A Secure Grid Medical Data Manager Interfaced to the gLite Middleware

Johan Montagnat, Akos Frohner, Daniel Jouvenot, Christophe Pera, Peter Kunszt, Birger Koblitz, Nuno Santos, Charles Loomis, Romain Texier, Diane Lingrand, et al.

A Secure Grid Medical Data Manager Interfaced to the gLite Middleware.

Johan Montagnat (johan@i3s.unice.fr)
CNRS, I3S laboratory, http://www.i3s.unice.fr/~johan

Ákos Frohner (akos.frohner@cern.ch)
CERN, http://www.cern.ch

Daniel Jouvenot (jouvenot@lal.in2p3.fr)
CNRS, LAL laboratory, http://www.lal.in2p3.fr

Christophe Pera (christophe.pera@creatis.insa-lyon.fr)
CNRS, CREATIS laboratory, http://www.creatis.insa-lyon.fr

Peter Kunszt (pkunszt@cscs.ch), Birger Koblitz (birger.koblitz@cern.ch) and Nuno Santos (nuno.santos@cern.ch)
CERN, http://www.cern.ch

Charles Loomis (charles.loomis@cern.ch)
CNRS, LAL laboratory, http://www.lal.in2p3.fr

Romain Texier (texier@polytech.unice.fr) and Diane Lingrand (lingrand@polytech.unice.fr)
CNRS, I3S laboratory, http://www.i3s.unice.fr

Patrick Guio (patrick.guio@bccs.uib.no)
University of Bergen, http://www.uib.no

Ricardo Brito Da Rocha (ricardo.rocha@cern.ch) and Antonio Sobreira de Almeida (antonio.sobreira.de.almeida@cern.ch)
CERN, http://www.cern.ch

Zoltán Farkas (zfarkas@sztaki.hu)
STA SZTAKI, http://www.sztaki.hu

Abstract. The medical community is producing and manipulating a tremendous volume of digital data for which computerized archiving, processing and analysis is needed. Grid infrastructures are promising for dealing with challenges arising in computerized medicine but the manipulation of medical data on such infrastructures faces both the problem of interconnecting medical information systems to grid middlewares and of preserving patients' privacy in a wide and distributed multi-user system. These constraints are often limiting the use of grids for manipulating sensitive medical data.

This paper describes our design of a medical data management system taking advantage of the advanced gLite data management services, developed in the context of the EGEE project, to fulfill the stringent needs of the medical community. It ensures medical data protection through strict data access control, anonymization and encryption. The multi-level access control provides the flexibility needed for

implementing complex medical use-cases. Data anonymization prevents the exposure of most sensitive data to unauthorized users, and data encryption guarantees data protection even when it is stored at remote sites. Moreover, the developed prototype provides a grid Storage Resource Manager (SRM) interface to standard medical DICOM servers thereby enabling transparent access to medical data without interfering with medical practice.

Keywords: Secure grid storage, gLite middleware, medical data management

1. Context

1.1. Objectives

Many scientific areas benefit from large and distributed storage capabilities provided by grid infrastructures. On top of physical storage resources, the EGEE [17] grid data management system eases the manipulation of large data volumes and provides high level functionality such as data distribution, replication and optimized access. To build a data management system that can adapt to the heterogeneous data storage resources (disk, tapes, silos...), the grid community has adopted standard interfaces to virtualize the underlying resources. In particular, gLite [23], the next generation EGEE middleware, has adopted the *Storage Resource Manager* (SRM) interface [29] standardized in the context of the Open Grid Forum [28]. The SRM's primary concern is to provide efficient access to large volumes of data. It provides, among other services, prefetching of data files recorded on secondary storage, management of storage space and reservation of storage resources. However, it does not provide any access control nor protection of data which severely limits its usability for applications manipulating sensitive data.

In this paper, we address the problem of sensitive data management on the EGEE grid infrastructure and we introduce a data management service designed to handle medical records on grids. We first motivate our approach through an in-depth requirement analysis of data management in the medical area (Section 2).

We then introduce a set of gLite services designed to implement secured data storage (Section 3) based on this analysis. They permit fine-grained access control for application data and metadata using standard grid credentials and they provide on-the-fly encryption/decryption capabilities.

We finally propose a software architecture to implement a *Medical Data Management* (MDM) service (Section 4). Our proposal provides access to medical sources for grid services and users while taking into account the constraints related to clinical practice. In particular pa-

tient privacy is guaranteed, data location is maintained in acquisition centers, and the standard clinical data flow is preserved. An implementation of these services interfaced to the EGEE middleware is finally demonstrated. Real medical use cases are discussed in Section 5 and performances are reported in Section 6.

1.2. Related Work

Building a grid interface to medical storage is of high interest for the medical imaging community and some systems have recently been reported. The TRENCADIS medical data manager [11] for instance is a WSRF-compliant middleware designed for sharing, searching and processing medical images and structured radiological reports. It is one of the first examples of a fully integrated grid system taking into account native medical image interfaces, security constraints and medical ontologies. Compared to the medical data manager presented in this paper, it is not interfaced to a distributed data management system and it can therefore not be used transparently by grid applications interfaced to the data management middleware.

The Storage Resource Broker middleware [8] is also used in the context of the BIRN [9] project for storing and processing large amounts of distributed medical data. It includes a distributed file management system and can manipulate metadata associated to files. However, it is not interfaced to native clinical storage unlike our system.

The security aspects concerning data access control and protection that are critical for medical systems have been thoroughly studied on distributed environments such as grids [30]. A consensus seems to emerge from the community to use shared secret key encryption algorithms for data protection [35, 34, 36, 38, 10]. They provide a robust and secure encryption framework. Data access control is a more controversial point. Role Based Access Control (RBAC) [21] is a flexible paradigm that can accommodate complex use cases as encountered in the medical area. Individual users are assigned roles, which represent collections of authorizations, depending on the tasks they are supposed to perform. Each user and each permission can be assigned to multiple roles. Many authorization expression languages (*e.g.* XACML [39]) and authorization assertion protocols (*e.g.* SAML [31]) have been proposed. In the context of the EGEE grid infrastructure, users are identified through X509 certificates. Roles are assigned to users through a Virtual Organization Membership Service (VOMS) [3]. The access control itself is implemented in various VOMS-aware data management systems based on Access Control Lists (ACLs).

2. Medical Data Management Requirements Analysis

2.1. Clinical Usage of Medical Data

The medical community is routinely using clinical images and associated medical data for diagnosis, intervention planning and therapy follow-up. Medical imagers are producing an increasing number of digital images for which computerized archiving, processing and analysis are needed [22, 27]. Indeed, image networks have become a critical component of the daily clinical practice over the years.

With their emergence, the need for standardized medical data formats and exchange procedures has grown [6]. For this reason, the *Digital Imaging and COmmunication in Medicine* standard [15] was adopted by a large consortium of medical device vendors. *Picture Archiving and Communication Systems* (PACS) [25], manipulating DICOM images and often other medical data in proprietary formats, are proposed by medical device vendors for managing clinical data. PACS are often proprietary solutions weakly standardized. PACS may be integrated to various degrees with the *Hospital Information System* (HIS) that holds administrative information about patients, and with the *Radiological Information Systems* (RIS) that holds additional information for the radiology departments. The DICOM standard, PACS, RIS and HIS have been developed with clinical needs in mind. They are easing the daily care of the patients and medical administrative procedures. However, their usage in other areas is very limited. The interface with computing infrastructures is almost completely lacking in particular. In addition, current PACS hardly address medical data management needs beyond clinical centers' administrative boundaries, while the patient medical folders are often spread widely over many medical sites that have been involved in the patient's healthcare. Many medical image acquisition devices weakly conform to the DICOM standard, thus barely hiding the heterogeneity of these systems.

2.1.1. *DICOM Format*

The DICOM standard encompasses, among other things, an image format and an image communication protocol. A DICOM image usually contains one slice (a 2D image) acquired using any medical imaging modality (MRI, CT-scan, PET, SPECT, ultrasound, X-ray... [1]). A DICOM image may contain a multi-slice data set but this is rarely encountered. A DICOM image contains both the image data itself and a set of additional information (or *metadata*) describing the image, the patient, the acquisition parameters and the radiology department. DICOM metadata are stored in fields. Each field is identified by a unique

tag defined in the DICOM standard. A given field may be present or absent depending on the imager that produced the image. The standard is open and image device manufacturers tend to use their own fields for various kinds of information. A couple of fields (such as image size) are mandatory but experience proved that surprises should be expected when analyzing a DICOM image. The image itself is usually stored as raw data. Most imaging devices produce one intensity value per image pixel, coded in a 12 bit format. Other formats may be encountered such as 16 bit data or lossless JPEG.

2.1.2. *DICOM Protocol, Storage, and Security*
Most (reasonably modern) medical image acquisition devices are DICOM clients. DICOM servers are computers with on-disk and/or tape back-ends able to store and retrieve DICOM images. The DICOM protocol defines the communication protocol between DICOM servers and clients.

The DICOM standard defines in particular a medical data format (part 10), messages and protocol (part 7) and security management profiles (part 15). The data format ensures independence from the DICOM application and media storage. However, it does not define any storage policy. DICOM servers implement their own policy of data storage. Data can be sent or retrieved from a DICOM compliant server using the DICOM protocol and standard messages to *push* images to the server or to *get* them from the server. The DICOM standard does not address issues of security policies although it mentions that "clearly adherence to appropriate security policies is necessary for any level of security". The standard only provides mechanisms that could be used to implement security policies with regard to the interchange of DICOM objects between DICOM-compliant applications. It is up to applications to agree on the desired level of security.

One should not see a DICOM data set as a set of files. As stated above, a single DICOM image usually contains only one image slice. In practice, during a medical examination (a DICOM *study*), a radiologist acquires several 2D and 3D images, comprising hundreds or even thousands of slices. A study is divided into one or several *series* and each series is composed by a set of slices (that can be stacked to assemble a volume when they belong to the same 3D image). Each slice is uniquely identified in its series by a unique *SOP* instance identifier. Note that there is often no notion of 3D image encoded in the DICOM format: a series may contain a set of slices composing several 3D images. The way a DICOM server stores these data sets on disk is irrelevant just like the way a database stores its table is usually not known by the users: the medical user is never exposed to the DICOM storage and does not

need to know if different files are used for each DICOM slice, series, study, etc. Metadata is included in DICOM image headers, making it difficult to search and manipulate. A DICOM server will often extract the images' metadata and store it in a database to ease data search.

The DICOM security model is weakened by its flexibility. It provides technical means of implementing security (relying on existing standards for data encryption) but it does not enforce any security policy. It is up to DICOM application to negotiate a level of security agreement. DICOM files may well be stored and transported unencrypted, exposing sensitive data. The DICOM server security model is based on a per-application basis: all users having access to some DICOM client application can access the information that the server exposes to this specific application. DICOM servers are using random file names without any connection to the patient information and a proprietary data storage policy. To cope with these data protection limitations, security is often implemented in hospitals by isolating the image network from the outside world.

2.1.3. *Access to Medical Images*

Each image acquisition device is a potential DICOM compliant medical image source. In a radiology department, one or several DICOM servers can be set up to centralize data acquired on this site. Medical data are naturally distributed over the different acquisition sites.

In clinical practice, physicians do not access directly image files. They identify data by associated metadata such as patient name, acquisition date and radiologist name. The data is transferred mainly for visualization purposes. The physician quickly scans the slices stack in the DICOM study and focuses on the slices he or she is interested in.

2.2. Medical Image Analysis

During the previous decades, with the growing availability of digital medical data, many medical data processing and analysis algorithms have been developed, enabling computerized medical applications that benefit of the patient and healthcare practitioners.

Although sharing the same data sources, the medical image analysis community has different requirements for medical systems than the healthcare community. Many algorithms are developed for processing and producing 3D image files: DICOM slices should be extracted from the DICOM server and assembled in a single volume before being processed as a coherent 3D data set. 3D images are exported to disk files for post-processing and ease of use. Various 3D medical image formats may be used to stack different DICOM slices into a single image volume

(the most common being the *analyze* file format). Images are also often identified through metadata, although the queries are not necessarily for nominative data but often related to the acquisition type or body region. Given the heterogeneity of image servers and 3D file formats, a common procedure for accessing all medical data sources is often needed.

2.2.1. *Medical Image Processing on Grid Infrastructures*

Given the enormous amount of medical data produced inside hospitals and the cost of medical data computing (especially image analysis algorithms), grids have proved to be very useful infrastructures for a large variety of medical applications [26]. Grids are providing computing resources and workload systems that ease application code deployment and usage. Moreover, grids are providing distributed data management services that are well suited for handling medical data geographically spread throughout various medical centers [13, 20, 12, 24, 7]. However, existing grid middlewares often only deal with data files and do not provide higher-level services for manipulating medical data. Medical data often has to be manually transferred and transformed from hospital sources to grid storage before being processed and analyzed. Such manual interventions are tedious and limit systematic use of grid infrastructures: some medical applications such as statistical atlas constructions or epidemiological studies may require the manipulation of thousands of images. In these cases, manual intervention is intractable due to the size of the data sets and the use of grids is only possible if medical images registration is automatically performed.

2.2.2. *Enforcing Medical Data Security*

Regulations applicable to medical records in European countries (and many others) enforce an extremely strict access control on medical data and patients privacy protection. In clinical practice, this is often obtained by isolating the image network from the external world. Management of medical data and deployment of private medical data processing applications on grids cannot be envisaged without demonstrating a very high level of security compliant with the current standards and practices of clinical data management.

All medical data should be considered as sensitive to preserve patient privacy. Nominative medical records are of course the most critical data and therefore, no binding between nominative data and images should be possible for non-accredited users. Only physicians participating to a patient healthcare should have access to the data of this patient.

On a grid, data distribution aggravates the security problems. To ensure patient privacy, the header of all DICOM images sent by a

DICOM server should be removed, at least partially, to ensure patients' anonymity. All images stored out of the source center should be encrypted to ensure that non-accredited users cannot read the image content.

2.2.3. *Bridging clinical data management systems with the EGEE grid middleware*

The first key to the success of the systematic deployment of medical image processing algorithms is to provide a data manager that:

- Provides access to medical data sources for computing without interfering with the clinical practice.
- Ensures transparency so that accessing medical data does not require any specific user intervention.
- Ensures a high level of data protection to respect patients privacy and data confidentiality constraints. The data protection scheme needs to be flexible enough to accommodate complex medical data manipulation scenarios involving many actors.

The next sessions describe a Medical Data Management system that we designed to ease medical data access and processing on a grid. It is compliant with DICOM clinical servers on the one side and interfaced to the EGEE grid data management system on the other side.

3. Secure Data Storage

3.1. EGEE Data Management System

The EGEE project is currently using the gLite middleware [23] on its production infrastructure. gLite is based on GLOBUS2, Condor, and other services developed in the European DataGrid project [16]. Our Medical Data Manager service (MDM) is strongly interconnected to gLite.

The gLite middleware provides workload management services for submitting computing tasks to the grid infrastructure and data management services for managing distributed files. The data management is based on a set of *Storage Elements* which are storage resources distributed by the various sites participating in the infrastructure (currently, more than 190 sites distributed all over Europe and beyond). All storage elements expose a common interface for interacting with the other middleware services: the *Storage Resource Manager* interface (SRM) that is standardized in the context of the Open Grid Forum.

The SRM is handling local data at the file level. It offers an interface to create, fetch, pin, or destroy files among other things. It does not implement data transfer by itself. Additional services such as GridFTP [4] or gLiteIO [19] are coexisting within storage elements to provide transfer and data access capabilities.

3.2. Data Distribution and Access Control

In addition to storage resources, the gLite data management system includes a *File Catalog* (FiReMAN - File Replication MANager [18]) offering a unique entry point for files distributed on all grid storage elements. Each file is uniquely identified through a *Globally Unique IDentifier* (GUID). The file catalog contains tables associating GUIDs to file locations. For efficiency and fault tolerance reasons, files may be replicated on different sites. Thus, each GUID may be associated to several physical replicas of the same logical file located at different places and identified by different site-dependent *Storage URL* (SURL). To ease the manipulation by users, human readable *Logical File Names* (LFN) can be associated to each logical file (each GUID).

The FiReMAN file catalog also associates metadata to files registered. Some metadata is mandatory (system metadata such as file checksum). Additional metadata can be defined by the users for application needs. In particular, ACLs can be associated to each file. They identify in a fine-grained manner (per individual user and/or per group) the access rights of each file. These access rights cover a rich set of capabilities: they include file list, read and write capabilities, associated metadata read and write capabilities, and access control modification capability. Users are individually identified through their certificate *Distinguished Name* (DN) that is referred to in ACLs.

Files are accessed using gLiteIO [19]. gLiteIO is a client/server based data exchange service. The client, installed on any grid client host (user interface or worker nodes) is the entry point to the data management system. It receives file access requests and contacts a gLiteIO server. The gLiteIO server then orchestrates the request. It first contacts the FiReMAN file catalog to check file access permissions and identifies the physical server concerned by the request. It then manages the file transfer. The gLiteIO server, being the only entry point to the data management system, ensures that all data access is properly authorized.

3.3. Data Encryption

gLiteIO is also the proper place to orchestrate additional services such as on-the-fly data encryption/decryption. It uses the standard symmetric key algorithm AES cipher available in the OpenSSL distribution for

that purpose. AES is thoroughly analyzed and recognized robust by the security community.

The encryption keys are stored in a specific set of servers named *Hydra*. Hydra provides controlled access to the encryption key (through certificate DN-based ACLs, similarly to what is done for files) and secured communication to the requester. In addition, Hydra exploits the Shamir secret sharing scheme [37] to improve security and reliability of this service. Shamir's scheme consists in splitting keys into n fragments stored in different places. Only m (with $m < n$) fragments are needed to reconstruct a complete key. However, owning less than m key fragments does not give any information on the complete key. Thus, the system is both resistant to attacks (at least m key stores need to be compromised for an attacker to be able to reconstruct a key) and reliable (the disconnection or loss of a limited number of servers does not prevent the key reconstruction). The Hydra servers hosting the key shares are completely identical in terms of interface and functionality.

In addition, gLiteIO may be parametrized to enable or disable data and control flows encryption independently. In our setting, data flow encryption is deactivated since data files are encrypted for storage and transported encrypted already. This avoids introducing an extra overhead with network encryption. Conversely, the control flow is encrypted so that user requests are not exposed.

3.4. Secure Metadata Manipulation

Patient records, containing personal data, are the most sensitive type of medical data. For this reason, metadata attached to medical images should be controlled at least as severely as the image data itself. Metadata needs to be stored in a database rather than in the image file headers both to benefit from the database system's fine-grained access control over the data and to improve the flexibility and performance of queries. The gLite AMGA [5] service is a secured front-end to traditional databases back-ends (MySQL, Postgres, Oracle). AMGA implements the gLite standard interface to metadata storage. It enables fine-grained access control through grid certificate with DN-based ACLs. Per table access control is provided and column access can be limited through specific table views. AMGA has been thoroughly tested and demonstrates excellent performance [32].

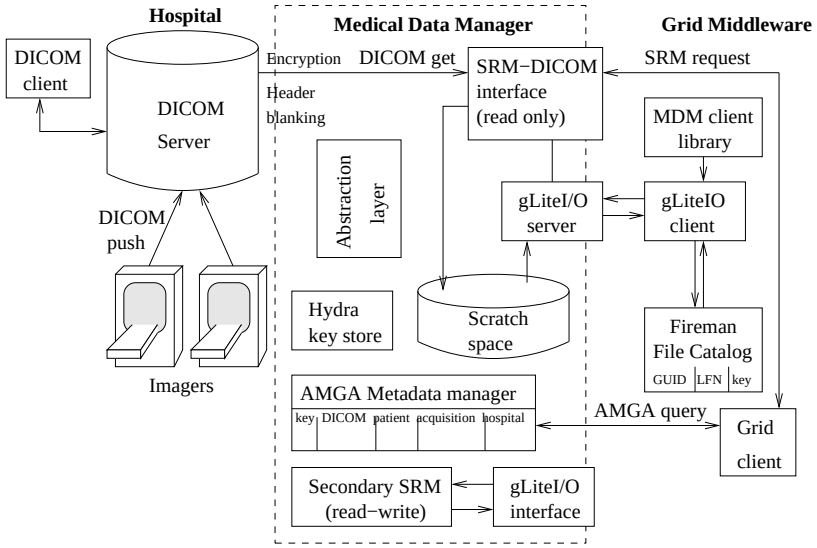

Figure 1. Overview of the medical data manager

4. Medical Data Management Service

4.1. Medical Data Manager Architecture

The Medical Data Management service architecture is diagrammed in Figure 1. On the left, the diagram shows a clinical site: various imagers in an hospital are *pushing* the images produced to a DICOM server. Inside the hospital, clinicians can access the DICOM server content through DICOM clients. In the center of Figure 1, the MDM internal logic is represented. On the right side, the grid services interfacing with the MDM are shown.

All middleware services requiring access to data storage do so through SRM requests sent to storage elements. To remain compatible with the rest of the grid infrastructure, our MDM service is based on a SRM-DICOM interface software. The SRM-DICOM core receives SRM requests and transforms them into DICOM transactions addressed to the medical servers. Thus, medical data servers can be shared between clinicians (using the classical DICOM interface inside hospitals) and image analysis scientists (using the SRM-DICOM interface to access the same data bases) without interfering with the clinical practice. An internal scratch space is used to transform DICOM data into files that are accessible through data transfer and access services (GridFTP or gLiteIO).

A metadata manager is also used to extract DICOM headers information and ease data search. In clinical use cases, the client usually first resolves grid identifiers through a request on the clinical metadata associated to the images before accessing these files. The AMGA service is used for ensuring secure storage of these very sensitive data. The AMGA server holds a relation between each DICOM slice and the image metadata. For data encryption needs, a Hydra catalog holding encryption keys associated to files is used.

This specialized SRM is not providing a classical Read/Write interface to a storage element. A classical R/W storage element can symmetrically receive grid files to be stored or deliver archived files to the grid on request. In the MDM, the SRM interface only accepts registration requests coming internally from the hospital as described in section 4.2. To avoid interfering with the clinical data, external grid files are not permitted to be registered on the MDM storage space: only get requests are authorized from the grid side. If classical grid storage is desired (with write capability), a secondary, standard SRM can be installed on the same host.

An *abstraction layer* is also depicted on the diagram. Its role is to offer a higher level abstraction for accessing 3D images by associating all DICOM slices corresponding to a single volume. Indeed, most medical image processing applications are not manipulating 2D images independently but rather consider complete volumes. The abstraction layer is associating a single GUID to each volume. On a request for the volume associated to this GUID, all corresponding DICOM slices (all slices with the same DICOM study and DICOM series identifiers but different slice numbers) are transferred from the DICOM server. They are assembled in a single volume in the order of their slice number on scratch space using a 3D file format.

4.2. Internal Service Interaction Patterns

To fulfill its role, the MDM service needs to be notified when files are produced by the imagers and stored into the DICOM server. This notification triggers a file registration procedure that is depicted in Figure 2. The DICOM data triggering the operation is first stored into the hospital DICOM server as usual. The DICOM header is then analyzed to extract image identifying information. This DICOM ID is used to build a replica SURL as used by the FiReMAN file catalog to locate files. The SURL is registered into the File Catalog and a GUID associated to this data on the grid side. The other metadata extracted from the DICOM header is stored into the AMGA metadata server. Finally, an encryption key that is associated to the file and that will

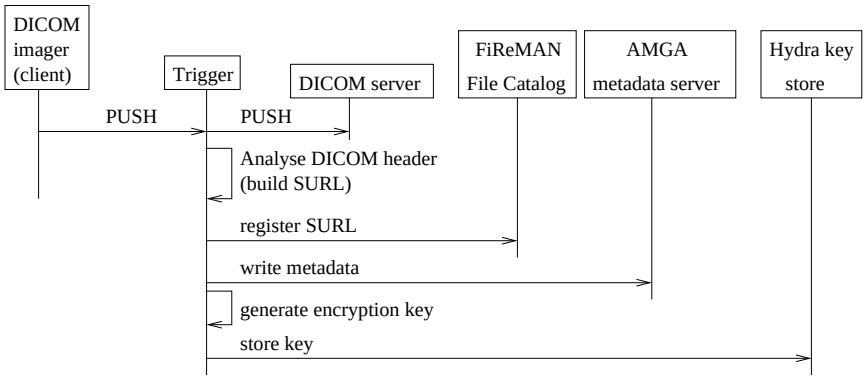

Figure 2. Triggered action at image creation

be used for data retrieval is generated and stored into the Hydra key store. The key associated to each file is unique and created at the file registration time.

Once DICOM data sets have been registered into the MDM, the server is able to deliver requested data to the grid as depicted in Figure 3. A client library is used for this purpose. To cover all application use cases, the MDM client library provides APIs for requesting files based on their grid identifier (GUID) or on the metadata attached to the file. In case of a request on the metadata, a database query is first made to the AMGA server and the list of GUIDs of images matching the query are returned. The client can then resolve the GUIDs associated to files matching the query and query the SRM-DICOM server. The gLiteIO server is the entry point to the storage manager for the client. It checks the client credential and controls the file access before resolving the physical file name. An SRM request can then be executed to retrieve the data from its SURL. SRM get requests are translated into DICOM get queries by the SRM-DICOM core. Data extracted from the DICOM server is first written to an internal scratch space. The image volume assembled is written using a simple 3D image file format (a human readable header including image size and encoding, followed by the raw image data). In this transformation the DICOM headers, which contains patient identifying information, are removed to preserve anonymity. The files are also encrypted before being sent out to ensure that no sensitive information is ever transferred or stored on the grid in a readable format. Files are then transferred through the gLiteIO service and returned to the client in an encrypted form. The file is only decrypted in memory of the client host, given that the client is authorized to access the file encryption keys.

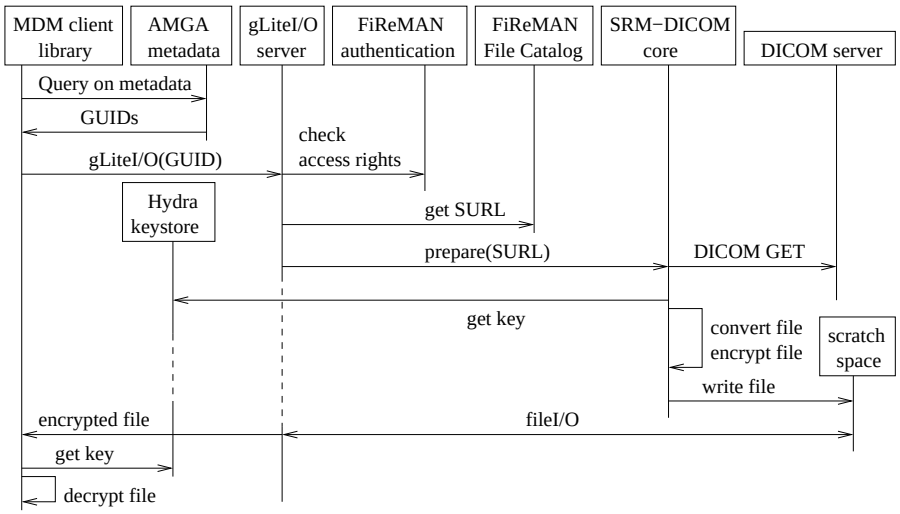

Figure 3. Accessing DICOM images

4.3. MDM Client

On the client side, three levels of interfaces are available to access and manipulate the data held by the MDM:

1. The MDM is seen from the middleware as any storage resource exposing a standard SRM interface, the standard data management client interface can be used to access images provided that their GUID is known. The files retrieved using this standard interface are encrypted.

2. The second interface is an extra middleware layer which encompasses access to the encryption key and the SRM. Thus images can be fetched and decrypted locally.

3. The third and last level of interface is the fully MDM aware client library represented in Figure 3. It provides access to encrypted files and in-memory decryption of the data on the application side, plus access to the metadata through the AMGA client interface.

These three access modes correspond to different data access use cases as illustrated in section 5.2.

5. Discussion

5.1. Data security

The security model of the MDM relies on several services:

1. file access control,
2. file anonymization,
3. file encryption, and
4. secure access to metadata.

The user is coherently identified through a single X509 certificate and all services involved in security use the same identification procedure. The file access control is enforced by the gLiteIO service which accepts ACLs for fine-grained access control. The Hydra key store and the AMGA metadata services also accept ACLs. To read the content of an image, a user needs to be authorized both to access the file and the encryption key. The access rights to the sensitive metadata associated with the files are managed independently. Thus, it is possible to grant access to an encrypted file only (*e.g.* for replicating a file without accessing to the content), to the file content (*e.g.* for processing the data without revealing the patient identity), or to the full file metadata (*e.g.* for medical usage).

5.2. Medical Application Use Cases

Through ACLs, it is possible to implement complex use cases, granting access rights (for listing, reading, or writing) to patients, physicians, healthcare practitioners, or researchers needing to process medical data, independently from each other. A typical scenario of medical image manipulation on a grid infrastructure from image acquisition to results storage is as follows:

1. The image of a patient is acquired in an hospital by a radiologist. The images is archived on the hospital DICOM server. The grid registration procedure is triggered: a GUID is registered in the file catalog and an encryption key is registered in the key store. Theoretically both the patient and the radiologist have access to this data: the patient might have read access while the radiologist will have read/write access (he may decide to delete unnecessary images, or to add image interpretation records). An hospital data administrator, responsible for archiving the image, should also get

R/W access to the image and associated metadata. The file catalog ACL can be updated with the patient DN (associated to the read capability), the radiologist DN (R/W capability) and the administrator DN (R/W capability) at image registration time. The key ACL will similarly be updated to enable access to the image content. Note that the SRM-DICOM server host itself should also be allowed to access the encryption key (read capability): the key will be needed for encrypting the file prior to any data transfer.

2. The physician who requested the acquisition will also need access to the image. The administrator can grant this access right by adding the physician DN to the data ACL (read capability). The access right can be provided to a single person or a group (the team of physicians working in the department). Given that physicians get full access right (to both data and metadata) of their patient, the read capability is set to the file ACL, the key ACL and the metadata ACL.

3. A scientist may need to access the data for a specific study involving data analysis. On request, the administrator can grant read access to the data file and encryption key for enabling access from the grid to this specific user. The patient metadata however do not need to be exposed to the scientist. If the image acquired is part of a specific medical protocol for which all images are accessible for scientific usage, a group authorized to access the images can be set up and the scientist added to this group.

4. Alternatively, an automatic backup mechanism might trigger files replication every night to provide recovery (in case of a main server failure) or high availability on the grid. This server might be authorized to access the file (read capability in the file ACL) but it does not need to access the file content nor the associated patient records (key and metadata ACLs are unchanged).

The system architecture described in this paper make such complex use cases possible while preserving the native medical data storage. Yet, the access to data for authorized grid users is completely transparent and does not require more effort than accessing any grid file.

5.3. Medical Metadata Schema

A minimal metadata schema is defined in the MDM service for all stored images. It provides basic information on the patient owning the image, the image properties and acquisition parameters. There are two main indexes used: a patient ID, for all nominative information associated to

patients and the image GUID for all information associated to images. The patient ID is a unique but irreversible field (such as a MD5 sum on the patient field name). Four main relational tables are used:

- The Patient table, indexed by the patient ID, contains the most sensitive identifying data (patient name, sex, date of birth, etc).
- The Image table, indexed by the image GUID, contains technical information about the image (size, encoding, etc). It establishes a relation with the patient ID.
- The Medical table, indexed by the image GUID, contains additional information on the acquisition (image modality, acquisition place and date, radiologists, etc).
- The DICOM table, indexed by the image GUID, contains the image DICOM identifiers used for querying the DICOM server.

To remain extensible, an additional Protocol table associates image GUIDs with medical protocol name. Through AMGA, the user can create as many medical protocols as needed, containing specific information related to some particular acquisition (*e.g.* a temporal protocol for cardiac acquisitions, etc). AMGA also enables per table access right control, allowing restricting access to the most sensitive data (*e.g.* the Patient table) to the minimum number of users.

In the current prototype, a centralized AMGA server is used to store all metadata segments extracted from the different images stored in several DICOM servers. A more realistic deployment scenario in clinical environment would be to distribute metadata over the different sites owning the data both for improving reliability (avoiding to create a single point of failure) and making the system more acceptable to the end users (sensitive information remain stored inside the hospital). This evolution will be helped by the federation mode proposed in the latest AMGA version [33].

5.4. Access Control Policies

The current system only provides a low level interface to control the access (*i.e.* manage the ACLs associated) to data and metadata. FiReMAN and AMGA both provides their own ACL management interface independently. In the future, the MDM should provide a high level interface to control access rights granted to the system users to the different part of data and metadata. This system should take into account the metadata distribution over the different acquisition centers. Different institutions will want to implement different access control

policies to their data for different user groups. In particular, physicians should only get access to metadata of their patients. Defining and implementing multiple access control policies is a difficult problem that requires RBAC technologies [21].

6. Deployment and Results

6.1. Testbed

The Medical Data Manager has been deployed on several sites for testing purposes. Three sites are actually holding data in three DICOM servers installed at I3S (Sophia Antipolis, France), LAL (Orsay, France) and CREATIS (Lyon, France). In the current testing phase, the MDM is not interfaced directly to clinical servers. We have installed a *Central Test Node* DICOM server [14] at each site. CTN is a free DICOM server implemented by the Mallinckrodt Institute of Radiology. It was designed to be used at the Radiological Society of North America annual meetings to foster cooperative demonstrations by the medical imaging vendors. The goal is to provide a centralized implementation that facilitates DICOM systems interoperability tests. In a clinical environment, any other DICOM server implementation could be substituted. In addition to the DICOM servers, these sites have installed the core MDM services: a SRM-DICOM server and the associated database back-end, a gLiteIO service, a GridFTP service, and all dependencies in the gLite middleware. Clients have been deployed on all these three sites.

To complete the installation, an AMGA catalog has also been set up in CREATIS (Lyon) for holding all sites' metadata. A FiReMAN file catalog for indexing files and a Hydra key store for keeping file encryption keys are deployed at CERN (Geneva, Switzerland).

Given the number of services involved, the installation and configuration procedure is currently complex. It is being simplified to ease the testbed extension. The MDM service should be deployed in hospitals where little support is provided for the informatics infrastructure.

The testbed deployed has been used to demonstrate the viability of the service by registering and retrieving DICOM files across sites. For testing purposes, DICOM data registrations are triggered manually. Registered files could be retrieved and used from EGEE grid nodes transparently, using the standard EGEE data management interface.

In a near future, the MDM service will be deployed on the EGEE grid infrastructure: the next generation production middleware (gLite 3.0), currently under deployment, encompasses all gLite clients needed to access this service, thus enabling access to an enlarged scientific

community. A longer term goal is to test the system in connection with hospitals by registering real clinical data freshly acquired and registered on the fly from the hospital imagers. This step involves entering a more complex clinical protocol with strong guarantee on the data privacy protection. The security cannot be neglected at any level at this point.

6.2. Performance Analysis

The most critical part of the MDM system in terms of performance is the image access. Therefore this was analyzed for performance bottlenecks. Two sets of experiments are reported in Figure 4. On the left column (red curves), the client used is located at I3S, close to the SRM-DICOM serving the image retrieved. On the right column (green curves), the client is located at CERN, close to the file catalog and the key store. In both test cases the same image file was retrieved 69 times from the server. The results are averages of the measured values to compensate for network and system load variability.

In these tests, a small image (approximately 400kB) was retrieved. This corresponds to a highly pessimistic case as a large part of the system overhead is independent from the image size. The total access time is in the order of 14s in both cases. The table in Figure 4 displays the average time for each step involved in the process (in seconds) and the fraction of the total transfer time that this represents. The curves at the bottom gives a graphical view of the times (left) and overhead fractions (right).

The steps represented correspond to the services calls depicted in Figure 3. The total time is measured by the client as the time spent between the image request and its availability on the client node. It is divided in three main steps:

1. **client - open** where the request is authorized, the file location is resolved via FiReMAN, the DICOM server is queried and the image volume is assembled and encrypted in scratch space prior to transfer.

2. **client - decryption init** where the client fetches the decryption key and initializes the decryption context. It includes the time spent by the server to service the demand and the round-trip time to receive the key.

3. **client - read** where the client actually reads the blocks of the encrypted file via remote I/O and decrypts them locally.

The rest of the time spent in the system is summarized in a fourth and last step labeled 'other'.

	I3S	CERN
client - open	7.64 [55.8%]	8.27 [58.8%]
gLiteI/O – access control	0.65 [4.7%]	0.65 [4.6%]
gLiteI/O – resolve SURL	0.69 [5.1%]	0.65 [4.7%]
gLiteI/O – SRM get	5.51 [40.3%]	5.60 [39.8%]
SRM-DICOM — retrieve key	0.73 [5.3%]	0.70 [5.0%]
SRM-DICOM — DICOM transaction	4.79 [35.0%]	4.90 [34.8%]
gLiteI/O – open overhead	0.78 [5.7%]	1.37 [9.7%]
client - decryption init	2.82 [20.6%]	1.35 [9.6%]
client – retrieve key	0.74 [5.4%]	0.40 [2.8%]
client – init overhead	2.08 [15.2%]	0.95 [6.7%]
client - read	0.13 [1.0%]	1.37 [9.7%]
other	3.08 [22.6%]	3.08 [21.9%]
total	13.68 [100.0%]	14.08 [100.0%]

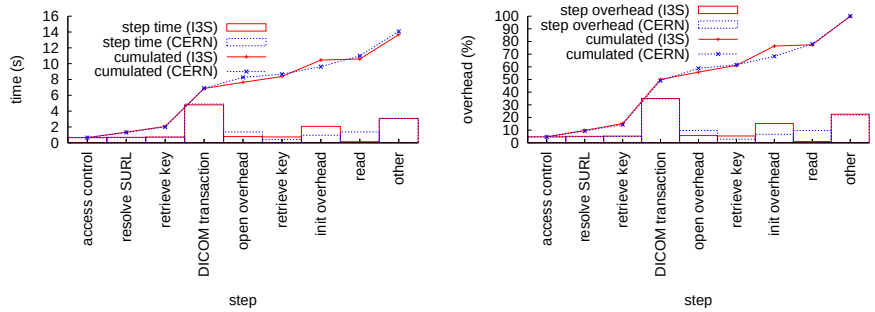

Figure 4. File access performances. Top: access time from I3S and CERN. Bottom: Graphical representations of access time (bottom left) and overhead (bottom right).

The access to the DICOM storage depends on the DICOM server installed and is not under our control. Using the CTN server installed for our experiments, the DICOM transaction represents approximately 35% of the total time. The remaining 65% are due to the system overhead (security pay off, network transmission). In case of a larger file, the relative time spent for DICOM transaction would increase as a significant fraction of the system overhead is independent from the file size.

The numbers plotted in Figure 4 show that the main differences between the clients located at I3S (close to the data) and CERN (close to the catalogs) are measured for security context initialization and for

data transfer. Indeed, it took about 20% of the total time to retrieve the encryption key and to initiate the security context at I3S against 10% at CERN, while it took about 10% of time to transfer data to CERN against 1% at I3S. The rest of system overheads is similar for both clients. This proportions would vary with different size files (the encryption key access time would remain almost constant while the file transfer time would vary).

It appears that the latency of remote calls, especially to a different site, is significant (more than 0.5s [5%] each). Thus the number of these calls should be minimized. The security context initialization, in particular, causes a noticeable overhead, larger than the data transfer itself even when considering a remote site. Although remote calls to security services are mandatory for dealing with medical data, dealing with full volumes rather than individual slices could drastically reduce the number of such communications (a single authentication is needed for retrieving a volume). To go further in that direction, bulk operations could be envisaged.

7. Conclusion and Perspectives

The Medical Data Manager service presented in this paper is an important milestone for enabling medical image processing applications on a grid infrastructure. Its main strengths are:

- To access medical databases without interfering with clinical practice. Data are kept on clinical sites and transparently transferred to the grid only when needed.
- To expose standard interfaces to other grid services. The MDM is fully integrated into the gLite middleware. Hence it benefits from the data distribution and replication capabilities provided by the data management system.
- To ensure a high level of security to preserve patients privacy.

The MDM prototype was successfully deployed and tested in a controlled environment. The next step is to interface to medical imagers inside hospitals. It will require to simplify the installation and configuration procedures as much as possible.

The core MDM development is not finished yet and additional functionality will be included to enrich the service. In particular, metadata is expected to be distributed in the different clinical sites where data is acquired rather than being centralized as it is the case in our testbed.

This configuration will be more acceptable to the clinical data administrators who want to keep control on the hospital data. It will require deploying several AMGA servers on different sites and exposing a centralized query service able to retrieve data from these different servers.

Acknowledgments

This work is partially funded by the EGEE contract [17] and the AGIR project [2]. We are grateful to the EGEE staff for providing support to this service development.

References

1. Acharya, R., R. Wasserman, J. Sevens, and C. Hinojosa: 1995, 'Biomedical Imaging Modalities: a Tutorial'. *Computerized Medical Imaging and Graphics (CMIG)* **19**(1), 3–25.
2. AGIR: project from the French research program "ACI Masse de données", Global Analysis of Radiological Images: `http://www.aci-agir.org/`.
3. Alfieri, R., R. Cecchini, V. Ciaschini, L. dell'Agnello, Á. Frohner, A. Gianoli, K. Lörentey, and F. Spataro: 2003, 'VOMS, an Authorization System for Virtual Organizations'. In: *European Across Grids Conference (EAGC)*.
4. Allcock, B., J. Bester, J. Bresnahan, A. Chervenak, I. Foster, C. Kesselman, S. Meder, V. Nefedova, D. Quesnal, and S. Tuecke: 2002, 'Data Management and Transfer in High Performance Computational Grid Environments'. *Parallel Computing Journal (PCJ)* **28**(5), 749–771.
5. AMGA: ARDA metadata catalog project: `http://project-arda-dev.web.cern.ch/project-arda-dev/metadata/`.
6. Andriole, K., R. Morin, R. Arenson, J. Carrino, B. Erickson, S. Horii, D. Piraino, B. Reiner, J. Seibert, and E. Siegel: 2004, 'Addressing the Coming Radiology Crisis: The Society for Computer Applications in Radiology SCAR Transforming the Radiological Interpretation Process (TRIP) initiative'. *Journal of Digital Imaging (JDI)* **17**(4), 235–243.
7. Barillot, C., R. Valabregue, J. Matsumoto, F. Aubry, H. Benali, Y. Cointepas, O. Dameron, M. Dojat, E. Duchesnay, B. Gibaud, S. Kinkingnéhun, D. Papadopoulos, M. Pélégrini-Issac, and E. Simon: 2004, 'NeuroBase: Management of Distributed and Heterogeneous Information Sources inNeuroimaging'. In: *Distributed Database and processing in Medical Image Computing workshop (DiDaMIC'04)*. Saint Malo, France.
8. Baru, C., R. Moore, A. Rajasekar, and M. Wan: 1998, 'The SDSC Storage Resource Broker'. In: *IBM Center for Advanced Studies Conference (CASCON'98)*. Toronto, Canada.
9. BIRN: Biomedical Informatics Research Network: `http://www.nbirn.net/`.
10. Blanchet, C., R. Mollon, and G. Deléage: 2006, 'Secured distributed service to manage biological data on the EGEE Grid'. In: *HealthGrid'06*. Valencia, Spain, pp. 142–152.

11. Blanquer Espert, I., V. Hernández García, and D. Segrelles: 2006, 'TRENCADIS - A WSRF Grid Middleware for Mnaging DICOM Structured Reporting Objects'. In: *HealthGrid'06*. Valencia, Spain, pp. 381–391.
12. Blanquer Espert, I., V. Hernández García, and J. Segrelles Quilis: 2005, 'Creating Virtual Storages and Searching DICOM Medical Images through a GRID Middleware based in OGSA'. *Journal of Clinical Monitoring and Computing* **19**(4-5), 295–305.
13. Budgen, D., M. Turner, I. Kotsiopoulos, F. Zhu, K. Bennett, P. Brereton, J. Keane, P. Layzell, M. Russell, and M. Rigby: 2005, 'Managing healthcare information: the role of the broker'. In: *HealthGrid'05*. Oxford, UK.
14. CTN: Central Test Node: `http://wuerlim.wustl.edu/DICOM/ctn.html`.
15. DICOM: Digital Imaging and COmmunication in Medicine: `http://medical.nema.org/`.
16. EDG: DataGrid FP5 European project: `http://www.edg.org`.
17. EGEE: Enabling Grids for E-sciencE phase I and II, FP6 European IST project, contract number INFSO-RI-508833: `http://www.eu-egee.org/`.
18. EGEE JRA1: 2005a, 'Fireman Catalog User Guide'. JRA1 Data Management Cluster. `https://edms.cern.ch/document/570780`.
19. EGEE JRA1: 2005b, 'gLite I/O User Guide'. JRA1 Data Management Cluster. `https://edms.cern.ch/document/570771`.
20. Ellisman, M., C. Baru, J. Grethe, A. Gupta, M. James, B. Ludäscher, M. Martone, P. Papadopoulos, S. Peltier, A. Rajasekar, S. Santini, and I. Zaslavsky: 2005, 'Biomedical Informatics Research Network: An Overview'. In: *HealthGrid'05*. Oxford, UK.
21. Ferraiolo, D. and D. Kuhn: 1992, 'Role Based Access Control'. In: *NIST-NCSC National Computer Security Conference*. pp. 554–563.
22. Germain, C., V. Breton, P. Clarysse, Y. Gaudeau, T. Glatard, E. Jeannot, Y. Legré, C. Loomis, I. Magnin, J. Montagnat, J.-M. Moureaux, A. Osorio, X. Pennec, and R. Texier: 2005, 'Grid-enabling medical image analysis'. *Journal of Clinical Monitoring and Computing* **19**(4–5), 339–349.
23. gLite middleware: `http://www.glite.org/`.
24. Hastings, S., S. Oster, S. Langella, T. Kurc, T. Pan, U. Catalyurek, and J. Saltz: 2005, 'A Grid-based image archival and analysis system'. *Journal of the American Medical Informatics Association (JAMIA)* **12**, 286–295.
25. Huang, H.: 1996, *PACS: Picture Archiving and Communication Systems in Biomedical Imaging*. Hardcover.
26. Montagnat, J., F. Bellet, H. Benoit-Cattin, V. Breton, L. Brunie, H. Duque, Y. Legré, I. Magnin, L. Maigne, S. Miguet, J.-M. Pierson, L. Seitz, and T. Tweed: 2004, 'Medical images simulation, storage, and processing on the european datagrid testbed'. *Journal of Grid Computing (JGC)* **2**(4), 387–400.
27. Montagnat, J., V. Breton, and I. Magnin: 2003, 'Using grid technologies to face medical image analysis challenges'. In: *Biogrid'03, proceedings of the IEEE CCGrid03 (Biogrid'03)*. Tokyo, Japan, pp. 588–593.
28. OGF: Open Grid Forum: `http://www.ogf.org`.
29. Perelmutov, T., D. Petravick, J. Gu, O. Barring, J.-P. Baud, S. De Witt, J. Jensen, O. Synge, M. Haddox-Schatz, B. Hess, A. Kowalski, and C. Watson: 2002, 'SRM Interface Specification v2.2'. Technical report, FNAL, USA.
30. Power, D., E. Politou, M. Slaymaker, and A. Simpson: 2006, 'Securing web services for deployment in health grids'. *Future Generation Computer Systems* **22**(5), 547–570.
31. SAML: `http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=security`.

32. Santos, N. and B. Koblitz: 2005, 'Metadata services on the grid'. In: *Advanced Computing and Analysis Techniques (ACAT'05)*.
33. Santos, N. and B. Koblitz: 2006, 'Distributed Metadata with the AMGA Metadata Catalogue'. In: *Workshop on Next-Generation Distributed Data Management (HPDC'06)*. Paris, France.
34. Seitz, L.: 2005, 'Conception et mise en oeuvre de mécanismes sécurisés d'échange de données confidentielles; application à la gestion de données biomédicales dans le cadre d'architectures de grille de calcul/données'. Ph.D. thesis, INSA, Lyon, France.
35. Seitz, L., J.-M. Pierson, and L. Brunie: 2003, 'Key management for encrypted data storage in distributed systems'. In: *IEEE Security in Storage Workshop (SISW'03)*. Washington DC, USA.
36. Seitz, L., J.-M. Pierson, and L. Brunie: 2005, 'Encrypted Storage of Medical Data on a Grid'. *Methods of Information in Medicine (MIM)* **44**(2).
37. Shamir, A.: 1979, 'How to share a secret'. *Communications of the ACM* **22**, 612–613.
38. Torres, E., C. De Alfonso, I. Blanquer Espert, and V. Hernández García: 2006, 'Privacy protection in HealthGrid: distributing encryption management over the VO'. In: *HealthGrid'06*. Valencia, Spain, pp. 131–141.
39. XACML: `http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=xacml`.

Address for Offprints:
Johan Montagnat
EPU, RAINBOW
930 route des Colles
BP 145
06903 Sophia Antipolis Cedex
France
tel: +33 492 96 51 03
fax: +33 492 96 50 55
e-mail: johan@i3s.unice.fr