Cloud, Grid and High Performance Computing:

Emerging Applications

Emmanuel Udoh
Indiana Institute of Technology, USA

Information Science
REFERENCE

Senior Editorial Director: Kristin Klinger
Editorial Director: Lindsay Johnston
Director of Book Publications: Julia Mosemann
Acquisitions Editor: Erika Carter
Development Editor: Hannah Abelbeck
Production Editor: Sean Woznicki
Typesetters: Michael Brehm, Keith Glazewski, Milan Vracarich, Jr.
Print Coordinator: Jamie Snavely
Cover Design: Nick Newcomer

Published in the United States of America by
in (an imprint of IGI Global)
701 E. Chocolate Avenue
Hershey PA 17033
Tel: 717-533-8845
Fax: 717-533-8661
E-mail: cust@igi-global.com
Web site: http://www.igi-global.com

Library of Congress Cataloging-in-Publication Data
Cloud, grid and high performance computing: emerging applications / Emmanuel Udoh, editor.
p. cm.
Includes bibliographical references and index.
Summary: "This book offers new and established perspectives on architectures, services and the resulting impact of emerging computing technologies, including investigation of practical and theoretical issues in the related fields of grid, cloud, and high performance computing"--Provided by publisher.
ISBN 978-1-60960-603-9 (hardcover) -- ISBN 978-1-60960-604-6 (ebook) 1. Cloud computing. 2. Computational grids (Computer systems) 3. Software architecture. 4. Computer software--Development. I. Udoh, Emmanuel, 1960-
QA76.585.C586 2011
004.67'8--dc22
2011013282

British Cataloguing in Publication Data
A Cataloguing in Publication record for this book is available from the British Library.

Chapter 13
Trusted Data Management for Grid-Based Medical Applications

Guido J. van 't Noordende
University of Amsterdam, The Netherlands

Silvia D. Olabarriaga
Academic Medical Center - Amsterdam, The Netherlands

Matthijs R. Koot
University of Amsterdam, The Netherlands

Cees T.A.M. de Laat
University of Amsterdam, The Netherlands

ABSTRACT

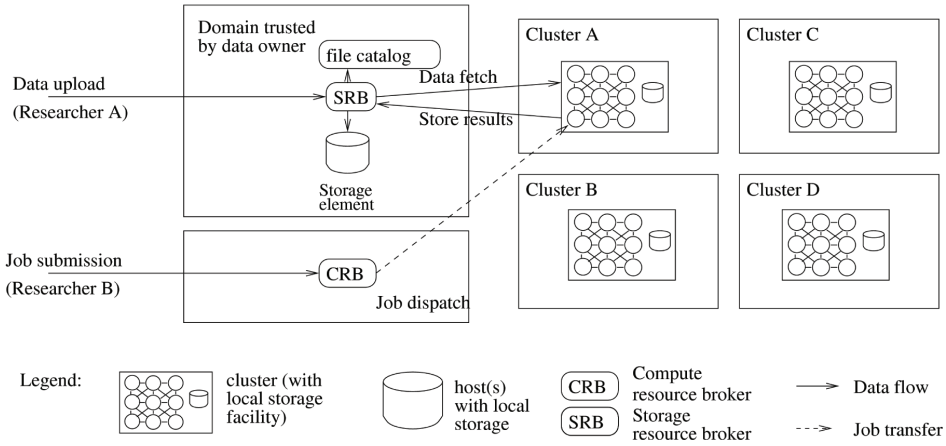
Existing Grid technology has been foremost designed with performance and scalability in mind. When using Grid infrastructure for medical applications, privacy and security considerations become paramount. Privacy aspects require a re-thinking of the design and implementation of common Grid middleware components. This chapter describes a novel security framework for handling privacy sensitive information on the Grid, and describes the privacy and security considerations which impacted its design.

INTRODUCTION

Most current Grid middleware is designed primarily for high-performance and high-throughput computing and data storage (LHC, n.d.; Foster, Kesselman, & Tuecke, 2001). Initially, Grid infrastructure aimed mostly at the Physics community, but recently many other domains, such as Biology, Pharmaceutics, and Medical research have shown increasing interest in using Grids for their applications. Grid middleware, including gLite (gLite, n.d.) and the Globus Toolkit (Globus, n.d.), hides many aspects such as data distribution and replication from users of the system. As a result, users are often unaware that jobs and data are transferred through multiple Grid components in different administrative domains implicitly. This makes it hard for users to understand the

DOI: 10.4018/978-1-60960-603-9.ch013

Figure 1. A use-case for medical imaging research showing grid resources in different administrative domains, with an emphasis on data and job flow



security implications of using Grid middleware, in particular when using it for applications that use privacy sensitive information.

Medical applications have very strict requirements on data handling and storage due to privacy concerns and regulations. Therefore, Grid middleware intended for usage in the medical domain should support policies that define where particular data may be stored, in what form, and what jobs from which users may access this data from what hosts or administrative domains.

This paper presents a new framework for managing privacy-sensitive data on the Grid, that allows for explicit data-owner control over data access and distribution related aspects. It makes a clear distinction between data storage components, access control, job authentication aspects, and auditing mechanisms for data related operations.

This paper is organized as follows: first we describe a use-case for medical research, based on our own experience (Olabarriaga, Nederveen, Snel & Belleman, 2006). Next, we analyze legal requirements with regard to medical data and technical aspects that are relevant when using Grid infrastructure to manage privacy-sensitive data. Finally, we describe a framework that allows data owners to express fine-grained data distribution and access control policies to allow for secure handling of medical data on the Grid. We conclude with an overview of some usability aspects.

USAGE SCENARIO

Figure 1 shows a typical Grid infrastructure deployment for medical research. A Grid storage system in one trusted administrative domain is used for storing medical research data. Although data is often replicated across different domains to enhance availability and reliability, we assume here that all storage facilities reside in only one administrative domain trusted by the data owner. Different incarnations of storage infrastructure exist, e.g., SDSC SRB and dCache (dCache, n.d.). In this paper, we refer to the storage infrastructure as a Storage Resource Broker (SRB) in a general way, without referring to a particular implementation.

First, Researcher A (data owner) uploads the data to an SRB he or she trusts, e.g., using gridFTP. Researcher B can now submit a job on the Grid through a Compute Resource Broker (CRB) which can reside in any administrative domain. The CRB transparently selects a cluster, typically based on load, where the job is scheduled for execution.

The user controls job submission via some job description, e.g., using a Job Submission Description Language (JSDL), which describes the binary to execute on the compute element and input files. In addition, the job description can specify a specific cluster, or resource requirements, to be matched with available Grid resources prior to scheduling. Running jobs can access files that the job's owner is authorized to access. In some cases, the Grid middleware pre-fetches required input files using the job's credentials prior to job execution.

Figure 1 also shows a File Catalog that provides a mapping between Grid 'logical file names' and the underlying physical files, which may be replicated on different storage systems on the Grid. Additionally, an SRB may also maintain a metadata service (not shown). Since metadata and file names may contain privacy sensitive information, both services should be managed by a trusted domain.

LEGAL REQUIREMENTS

The European Union (EU) has produced legislation on handling personal information and privacy (EC, 1995). This section focusses on EU and selected Dutch regulations. Countries outside the EU have adopted or are adopting legal measures to allow exchange of personal data with the EU countries (e.g., U.S. Safe Harbor Framework). For more information about other countries see (Fischer-Huebner, 2001; EC; Herveg, 2006; U.S. Congress, 1996).

EU regulations can be seen as leading guidelines for handling personal data (Fischer-Huebner, 2001). The data protection regulations can be summarized as follows. First, there must be a necessity for data collection and processing. Related to that, for each data collection, there has to be a clear purpose binding which specifies what is done with the information. Usage of data beyond this specified purpose is not allowed. In addition, a minimality principle exists, which states that only the minimum information for the required purpose may be collected. Furthermore, there has to be transparency of personal data processing and collection, implying that the data subject is informed of data collection (opt-in or opt-out) and that the data subject has a right to access the information. Finally, the regulations require that information is accurate, which implies that the information must be kept up-to-date.

Two Dutch laws (WGBO, 1994; WMO, 1998) formalize what may be done with data collected from a patient in the course of treatment. In general, usage of patient information outside the scope of the patient's treatment is not allowed, unless there is considerable public interest or similar necessity to do so. Medical scientific research is often considered such an exception (Herveg, 2006).

If a patient explicitly consents with usage of his data for medical research, that data is purpose-bound to a specific medical research activity. The data may not be disclosed beyond this activity. The physician or medical researcher who determines the purpose and means of processing is legally responsible for ensuring an appropriate level of security to protect data.

The restrictions described above only apply to personal data. In some situations, the data can be de-personalized to circumvent these restrictions, e.g., as done in (Kalra et al., 2005; Montagnat et al., 2007; Erberich et al., 2007). However, complete de-identification is hard to get right, and re-identification is often possible (Sweeney, 2002; Malin, 2002). For this reason, de-identified information should be considered confidential, and appropriate distribution and access control mechanisms are required.

BASIC GRID SECURITY INFRASTRUCTURE

The Grid Security Infrastructure (GSI) (Foster, Kesselman, Tsudik and Tuecke, 1998) is the de-

facto standard for user and host authentication on the Grid. GSI is used by most mature Grid middleware implementations. Shortcomings of this infrastructure are described later in this paper; here we introduce the basic GSI infrastructure.

GSI essentially comprises a Public Key Infrastructure (PKI) that is used to sign user identity and host certificates. Users can create limited-lifetime Proxy certificates which allow them to send credentials with their jobs for authentication, without the risk of compromising the user's private key. Proxy certificates are used for all transactions by a job, such as gridFTP transactions. We here assume that all authorization decisions with regard to data are based on GSI user authentication by means of Proxy certificates. Other approaches (such as role-based or attribute-based authorization, as proposed in (Alfieri et al., 2004) are possible, but not required for our framework. Many Grid infrastructures manage access control to resources and storage based on virtual organization (VO) membership information. However, VO-based authorization is often too course-grained for protecting medical information: there may be many users (e.g., researchers) in a VO, which may not all be equally trusted to access particular data. Therefore, we assume authorization based on user identities in this paper.

PROBLEM ANALYSIS

Grids are, by nature, distributed across multiple administrative domains, only a few of which may be trusted by a specific data owner. Grid middleware, and thus jobs, typically run on an operating system (OS), such as Linux, that allows administrators to access all information on the system. A job or data owner does not have control over the hardware or software that runs on some remote system. Besides OS and middleware vulnerabilities, these systems might also not be well protected against physical attacks, such as stealing hard disks. Such aspects should be part of a risk assessment when decisions are made on which sites are trusted to store or access particular information.

Given legal constraints, trust decisions will and should be conservative. For example, unencrypted data, file names, and other sensitive metadata should only be stored in trusted domains, e.g., in the hospital. This aspect is even more prevalent in systems where jobs on remote machines can access medical data. Current OSs such as Linux provide little assurance that information stored on the system cannot be leaked to external parties (van 't Noordende, Balogh, Hofman, Brazier and Tanenbaum, 2007).

Even if files are removed after the job exits (e.g., temporarily created files), the contents could be readable by administrators or possibly attackers while the job executes. Furthermore, disks may contain left-over information from a job's previous execution, which is readable by an attacker who gains physical access to a storage device, if the system is not properly configured (NIST). As another example, it is possible to encrypt swap space in a safe way, but this is an option that has to be explicitly enabled in the OS. For these reasons, it is important for a data owner to identify critical aspects of the administration and configuration of a remote host, before shipping data to (a job running on) that host.

Another problem is that a data owner cannot control nor know the trajectory that a job took before it was scheduled on a host, since this is implicit and hidden in current Grid middleware. Therefore, even if the host from which a job accesses data is trusted by the data owner, there is a risk that the job was manipulated on some earlier host.

Current middleware does not provide a way to securely bind jobs to Proxy certificates: a certificate or private key bundled with a program can easily be extracted and coupled to another program which pretends to be the original program. In Grids, this issue is exacerbated by the fact that a job may traverse several middleware processes

(e.g., a CRB) in different domains before it is scheduled at some host. Each of these hosts or domains may be malicious, and the administrator or an attacker that gains access to one of these hosts may replace the original job with another program that leaks information to an external party. Alternative authentication schemes (e.g., username/password-based) do not improve this situation.

For this paper, we assume that the implementation of a job is trusted when this job's owner is trusted. In particular, we assume that medical researchers are aware of confidentiality aspects regarding medical data and treat this data as confidential information – and as a result use only trusted programs to make use of this data. In the proposed framework, jobs can only access data from hosts that are trusted by the data owner, and we assume that a job submitted by a trusted user will not leak information to unauthorized parties. A mechanism is presented later in this paper that allow users to seal jobs in such a way that tampering with these jobs is not possible.

Note that mechanisms exist that limit the capabilities of a possibly untrusted program to export information to arbitrary external parties, e.g., using the jailing system described in (van 't Noordende, Balogh, Hofman, Brazier, and Tanenbaum, 2007). Such solutions can be considered as additional measures to increase security, but are outside the scope of this paper.

For this paper, we assume that jobs do not ship potentially privacy-sensitive (output) data back to the possibly untrusted CRB through which the they entered the system. Instead, jobs should be programmed to encrypt output data with the job owner's public key before returning to their CRB, or they should store any potentially sensitive (output) data only on secure storage, preferably the system that contained the input data.

Summarizing, a number of implementation issues should be solved before we can be sure that privacy-sensitive information cannot be accessed by unauthorized parties. First, a secure binding between jobs and Proxy certificates must be provided. Second, a data owner should be able to express in a policy which administrative domains he or she trusts to handle privacy sensitive information in a safe way, based on a risk assessment. Third, a data owner should be able to express policies with regard to a remote system's configuration details which are relevant to privacy and security and the way in which data is handled.

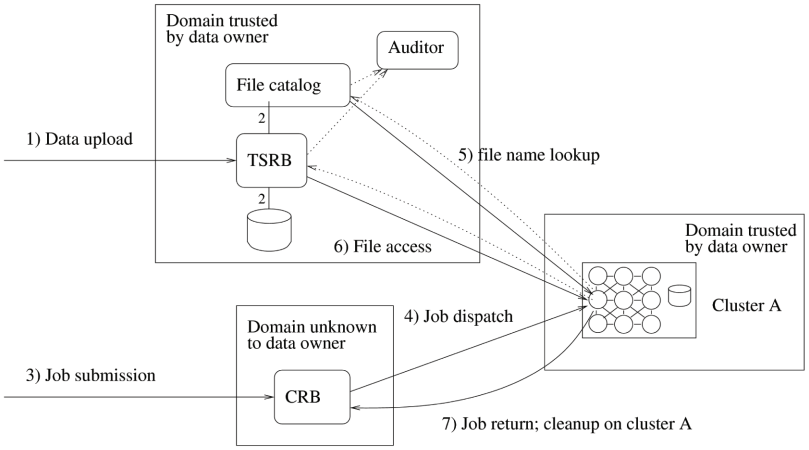
THE TSRB FRAMEWORK

We propose a framework for secure handling of privacy sensitive information on Grids that allows for controlling data access and distribution aspects. The components and interactions of the framework are presented in Figure 2.

The framework is centered around a secure storage infrastructure called Trusted Storage Resource Broker (TSRB). There may be many TSRBs on the Grid, possibly managed by different administrative domains in different VOs. The TSRB is coined ``trusted'', because (1) it is deployed in an administrative domain trusted by the data owner, and (2) it is trusted to enforce data-owner specified access control policies. The TSRB controls access to data items or collections by combining User-based Access Control Lists (User ACLs) and Host-ACLs. Host ACLs contain required host properties that must be met by a remote host before the data can be accessed by a job on this host.

Required host properties are described by the data owner in a Remote Host Property List (RHPL). Each host has a Host Property List (HPL) that contains host configuration details. The HPL contents are matched with the data's RHPL at connection time. The HPL is maintained by the remote host (Cluster A in Figure 2), and is signed by the host's administrator. The TSRB also maintains for each data collection or item a Host ACL containing a list of administrative domains

Figure 2. The TSRB framework: files, file names and metadata are managed by a Trusted SRB. Dotted lines depict microcontract establishment and auditing, solid lines depict data flow and job transfers



or hosts, who are trusted by the data owner both for confidentiality (of the administrators) and for providing correct information in their HPL.

The main actions are illustrated in Figure 2. A user uploads data to the TSRB, e.g., using gridFTP (step 1). The data is stored in a storage system maintained in the TSRB domain. Metadata can be stored in a separate service managed by the TSRB, e.g., a File Catalog in case of storing files (step 2). A job is submitted through a CRB (step 3), about which the data owner has no information. Eventually, the CRB submits the job to a cluster (step 4) that must be trusted by the data owner before the job can access data.

As part of the protocol before data access is authorized, user (job) and host authentication takes place, and the data's RHPL and the remote host's HPL are compared (details are given later). If RHPL and HPL match, a microcontract is established, which is a statement containing agreed-upon host properties and signed by both the TSRB and the remote host. Microcontracts are established for all authorization decisions, including, e.g., resolving file names in a File Catalog (step 5), and accessing the data item itself (step 6).

Only after the TSRB receives a microcontract, are the data shipped to the job or middleware acting on the job's behalf. In step 7 a job returns to its CRB where it can be collected by its owner. Subject to agreement in the microcontract, Cluster A ensures that no data from the job's execution remains on the host.

Auditing is important to allow data owners to track which jobs applied which operations on their data, on behalf of which users, and from which hosts. All established microcontracts are shipped to an auditor process (see Figure 2), which can be used by data owners to trace the transactions. Auditing can help establish trust (e.g., using reputation-based mechanisms), and enables tracking of potential sources of information leakage.

CONCEPTS AND INTERACTIONS

Job Authentication

A solution to provide a secure binding between jobs and Proxy certificates is to combine job integrity verification with a trust-based mechanism. Only if a data owner trusts a remote system to verify the integrity of incoming jobs properly, can he or she assume the the job-Proxy certificate binding to be valid, and can Proxy certificate-based au-

thentication be trusted. Job integrity verification can be implemented securely if all initial content of the job is signed by its owner, thus creating an unforgeable binding between all components of a job, including its proxy certificate.

A secure job container could be created before submitting the job, which is signed using the job owner's private key - see a similar idea in (van 't Noordende, Brazier & Tanenbaum, 2004). A job container has a well-defined structure, which makes it straightforward for the middleware to find the components of the job that are relevant for integrity verification. Alternative implementations are conceivable, e.g., using signed Virtual Machine images (Travostino et al., 2006).

Host Property Lists

For risk assessment and policy enforcement, hosts should announce security relevant properties of their operating system, its configuration, and the used middleware, including properties regarding job integrity verification, in their Host Property List (HPL). The host administrator has the responsibility to fill in the HPL correctly. As a concrete example, the HPL could report on whether the operating system was configured to use encrypted swap space, on whether the middleware is capable of job integrity verification, and provides jobs with a private file system that is removed after the job exits.

HPLs allow for run-time assessment on whether a host adheres to the requirements for secure data handling as imposed by a data owner. This assessment takes place at the time that a connection is made to the TSRB. Because HPL matching takes place at connection time, no external trusted repository of HPLs is required for security.

Microcontracts

Microcontracts state the obligations that the site holds with regard to a transaction. Our framework requires that all Grid middleware components that are concerned with data transfer aspects (e.g., gridFTP) are extended with functionality to report a signed HPL to their peer processes at connection time. Based on whether peers trust each other to provide correct information, and on the information in their HPLs, both parties decide whether to proceed with the transaction (e.g., data transfer), which takes place over a mutually authenticated secure channel. Agreement should be reached on the properties in the data item's RHPL *before* any data is shipped.

For *non-repudiation,* both parties must co-sign a microcontract once agreement is reached. Non-repudiation means that none of the parties can deny that they agreed on the contract's content. To allow for auditing the exact operations on a particular data item, the microcontract has to be bound to each individual transaction, by including e.g., a hash over the data and the operation in the microcontract.

Trusted Storage Resource Broker

The TSRB is the key component for managing all privacy sensitive data in our framework. The TSRB is the central reference monitor and access point for data stored through this TSRB. In particular, the TSRB enforces the access control policies outlined in this paper. For clarity of exposition, we assume that the TSRB is a non-distributed service running in a single domain. The TSRB (and by implication, domain) is determined as trusted by a data owner prior to storing data on it.

Although we refer to the TSRB as a resource broker here, the TSRB is effectively an abstraction for a secure storage system. In case where the TSRB uses distributed facilities (e.g., untrusted storage elements managed by different domains), the TSRB can implement broker functionality. In this case, the TSRB should make sure that it stores only encrypted data on untrusted storage, using cryptographic filenames. Example storage systems that are implemented as a broker for

encrypted data are described in (Montagnat et al., 2007; Xu, 2005).

Naming and Metadata Services

The TSRB can offer metadata services for managing and querying metadata about the stored data. Metadata is useful to search for data items of interest in large data collections. File names can be seen as metadata specific to file systems.

Naming or metadata services must be integrated into the TSRB, since access to file names and other sensitive metadata should be carefully protected. For example, careless encoding of file names could enable attackers to identify patient or hospital information from a file name and re-identify a patient. Naming or metadata services may be private to a VO, or part of some hierarchical naming service. In either case, file name lookup requests are subject to data-owner specified access control policies as outlined in this paper.

Access Control Lists

Access control in our system is enforced on the basis of ACLs. ACLs can be associated with individual data items or with a grouping (set) of data items. In case of files, grouping may be facilitated by e.g., associating ACLs with directory names. Unauthorized users should not even be able to find out if a given data item exists.

The User ACL contains a list of principals (job owners) that are allowed to access a (set of) data item(s), together with these principals' access rights on that data. The Host ACL specifies from what hosts or domains authorized jobs may access particular data, and with what access rights. Access rights from the User and Host ACLs are combined such that only the minimum set of rights for this data is granted to a job of a given user running on a given host.

The trusted domains or hosts in the Host ACL are determined by the data owner, e.g., based on whether he or she trusts the administrator of a particular administrative domain. Host ACLs are expressed as GSI host/domain name patterns, which match with the common name field of the x509 GSI host certificate, e.g., *.sara.nl, or host1.amc.nl. Specific patterns override wildcarded patterns. Also associated with data items or sets of data is a Remote Host Property List (RHPL). Before evaluating a remote host's HPL, it is checked that this host is in the Host ACL; only then is the HPL information considered trusted.

We chose to separately store an RHPL with each (set of) data items, in addition to the basic User and Host ACLs, because of the dynamic nature of Grid systems. Different domains may contain many machines or clusters, each of which with different configuration and job or data handling properties, which may even change over time. Connection-time RHPL / HPL matching allows the system to evaluate these properties at runtime, without relying on a (trusted) central repository of these properties.

Job Submission Procedure

At job submission time, a host must be selected from which the job's input data is accessible. Since CRBs are generally not trusted[1], client-side software should be used which contacts the TSRB before job submission. A file naming convention combined with a naming service (e.g., DNS) allows the client job submission program to locate the TSRB where the data is stored.

Client-side software can authenticate directly to the TSRB using the job owner's identity key. If authorized, it can fetch the relevant access control and HPL information, using which a job description is created. To allow for selection of suitable hosts by the CRB, HPLs could be published in a (global) information system. Note that because of run-time (R)HPL evaluation, the information system does not need to be completely consistent or trusted. This is important for scalability, as keeping a possibly global information system fully up-to-date may be infeasible.

Auditing

Auditing is important to allow for tracing all operations on a particular data item. For convenience and scalability, we use a trusted auditor process per TSRB, managed by the TSRB. Copies of the co-signed microcontracts of all transactions are sent to and strored by the auditor. This allows the data owners to trace all transactions that involve a particular data item in a way that ensures non-repudiability.

PUTTING IT ALL TOGETHER

Authorization of a data access requires that the connecting job's owner is on the User ACL, that the host on which the connecting job runs is on the Host ACL, and that the properties in the RHPL match the properties in the connecting host's HPL. Authorization of a data request consists of the following steps, assuming GSI host/Proxy certificate based authentication.

•At connection time, the connecting process (either a job or middleware, in case of data pre-fetching) authenticates with the TSRB using the job's Proxy certificate, resulting in an authenticated and encrypted SSL/TLS channel.

•The information from the Proxy certificate is matched against the User ACL to see if access is allowed. If not, an error is returned that does not indicate whether the data exists or not.

•The TSRB and the connecting process engage in a protocol for matching RHPL and HPL properties. If the connecting process is the middleware (e.g., during data pre-fetch), it can directly sign the microcontract. If the connecting process is a job, it has to request its local middleware (using a runtime interface) to match the RHPL of the TSRB with the host's HPL, and to have it sign a microcontract on its behalf if these properties match. The microcontract includes the (hash over the public key of the) Proxy certificate of the job to which it was issued.

•The signature over the microcontract (shipped together with the GSI host certificate that was used for signing) is compared with the Host ACL, to see if the HPL information is trusted and if access is allowed from this host.

The above mechanisms suffice to establish the required combination of Host ACL and User ACL based authorization, together with obtaining a microcontract signed by the connecting host before the data is shipped. If all provided information matches the data owner's requirements, the data is shipped to the requesting job or middleware, and the microcontract is logged in the auditor process.

USABILITY

Determining an appropriate Host ACL and HPL specification may be difficult for non-technical data owners. However, system administrators who support users may define template (R)HPLs with basic properties that hosts must adhere to when running jobs that access sensitive information. Such templates may be provided with the client-side software used for data uploading, and may be adapted by data owners and/or local system administrators at the time of use. Similarly, local (VO) administrators may help by composing default lists of trusted domains for particular data types or groups of users. Such measures allow secure usage of the system by researchers without burdening them with too many details. Dynamic adaptation of RHPLs for long-term storage of data is an open issue that needs to be addressed.

CURRENT STATUS AND FUTURE WORK

We have implemented a proof-of-concept implementation of the TSRB framework based on a gridFTP server from the Globus toolkit. We extended the gsi-FTP server with an authenticated key-exchange protocol to authenticate the client

and establish a secure connection for data transfer; FTP commands were modified to include TSRB concepts such as HPL exchange and microcontracts. The resulting system's performance is as well as can be expected from a protocol that uses encryption to protect data transferred from server to client. Performance results are described in a separate report (Coca, 2011).

One of the more difficult issues to address when using our system, is how to decide whether a given system setup is secure. We have experimented with HPLs to describe various Linux systems. To determine a system's security, we used information obtained from the Common Vulnerability and Exposures (CVE) vulnerability database (http://cve.mitre.org), to locate potentially vulnerable packages on the system. A vulnerability score (Scarfone and Mell, 2009) is associated with each entry in the CVE database, which indicates the potential impact of a vulnerability on security of the system. However, Grid systems generally have different characteristics than desktop systems, for which the scoring method was devised.

Grid clusters are typically batch systems, and worker nodes within a cluster are usually not directly exposed to the Internet. Rather, the most important threats may originate from *within* the cluster, for example from malicious jobs that run concurrently with a job in the same cluster, or from jobs that compromised a machine some time earlier. We are currently studying whether the CVE-based vulnerability scoring can be adapted to Grid-specific characteristics. We are also studying ways to facilitate dynamic evalutation of HPL-based policies, such that users or administrators do not have to be overly burdened by (manually) updating policies or analyzing vulnerability reports to assess a system's security.

RELATED WORK

Montagnat et al. (2007) describe a Medical Data Manager (MDM) for DICOM images and associated metadata in a secure way. MDM is deployed inside hospitals, and provides read-only access to automatically de-identified DICOM images to grid jobs outside the hospital's domain. Data is encrypted before it becomes accessible to Grid jobs, so jobs must first acquire a key from a key store before they can access the data. However, MDM does not constrain from which hosts jobs may access the data or keys. MDM's reliance on automatic de-identification of DICOM headers may prove a vulnerability, e.g., in case of images which contain facial features of a patient as part of the binary data.

Globus MEDICUS (Erberich, Silverstein, Chervenak, Schuler, Nelson, & Kesselman, 2007) is an approach for sharing medical information (metadata and files) through Grid infrastructure. Encryption can be used to store information securely on untrusted storage elements in the Grid. One of the weak points of the system is that it does not clearly describe where the different components reside physically, i.e., what the trust model is. For example, metadata is stored in a meta catalog service which may be operated outside the hospital domain. In addition, the system depends on GSI for authentication, which makes the lack of a clear trust model even more worrisome.

Blancquer et al. (2009) describe an approach for managing encrypted medical data, building upon Hydra (Xu, 2005) and the ideas presented in Montagnat et al. (2007). The contribution of this approach is that key management and authorization are integrated with common Grid management concepts such as Virtual Organizations. However, like MDM and Hydra, the approach chosen by Blancquer et al. does not deal with the problem that the machine where the data is decrypted (by the job) may be compromised.

None of the related work considers trust in the hosts or clusters from which data are accessed, nor with the properties of the software running on these hosts.

DISCUSSION

We presented a trust-based security framework for Grid middleware that allows for enforcement of access control and data export policies for privacy-sensitive data. The framework proposes a Trusted SRB to manage data and enforce fine-grained access control policies on behalf of data owners. Access control policies combine user-based access control and trusted hosts lists with a runtime evaluation of properties of remote hosts from which jobs request data access. Microcontracts allow for establishing data handling agreements, and an auditing mechanism based on microcontracts allows for tracing all operations on the data.

The focus of this paper is on usage scenarios where Grid-based storage and data sharing is required. Our framework emphasizes data-owner specified user and host (property) based access control policies, to ensure that privacy sensitive information is only made accessible to authorized jobs running on hosts trusted by the data owner. This way, we can ensure that the data owner's requirements for secure data handling are met. More generally, we believe that the basic concepts presented in this paper, such as remote host property list evaluation, microcontracts, and auditing, can be of value for any distributed system or Grid middleware component in which precise control is required over where data or code may be distributed, and under what constraints.

ACKNOWLEDGMENT

We thank Oscar Koeroo, Dennis van Dok, and David Groep (NIKHEF) for valuable insight in the gLite-based VL-e infrastructure. Keith Cover (VU Medical Center) provided valuable information on privacy aspects of his job farming application. Berry Hoekstra and Niels Monen worked on a student project on HPLs and vulnerability scoring. Razvan Coca (UvA) is thanked for recent contributions to implementing the framework described in this paper. This work has been carried out as part of the Dutch research project Virtual Laboratory for e-Science (VL-e).

REFERENCES

Alfieri, R., Cecchini, R., Ciaschini, V., dell'Agnello, L., Frohner, A., Gianoli, A., et al. Spataro, F. (2004). Voms, an authorization system for virtual organizations. *European Across Grids Conference, LNCS 2970*, (pp. 33-40). Springer, 2004.

Blancquer, I., Hernández, V., Segrelles, D., & Torres, E. (2009). Enhancing privacy and authorization control scalability in the Grid through ontologies. *IEEE Transactions on Information Technology in Biomedicine*, *13*(1), 16–24. doi:10.1109/TITB.2008.2003369

Coca, R. (2011). Security enhancements of GridFTP:Description and Measurements. *Technical Report UVA-SNE-2011-01*, University of Amsterdam.

Dcache. (n.d.). *Dcache storage system*. Retrieved from http://www.dcache.org/

E.C. (1995). Directive 95/46/EC. *European commission data protection regulations overview page*. Retrieved from http://ec.europa.eu/justice_home/fsj/privacy/

Erberich, S., Silverstein, J. C., Chervenak, A., Schuler, R., Nelson, M. D., & Kesselman, C. (2007). Globus medicus - federation of dicom medical imaging devices into healthcare grids. *Studies in Health Technology and Informatics*, *126*, 269–278.

Fischer-Huebner, S. (2001). *IT-security and privacy: Design and use of privacy-enhancing security mechanisms*. New York, NY: Springer-Verlag.

Foster, I., Kesselman, C., Tsudik, G., & Tuecke, S. (1998). A security architecture for computational grids. *Proc. 5th ACM Conf. on Computer and Communication Security,* (pp. 83-92).

Foster, I., Kesselman, C., & Tuecke, S. (2001). The anatomy of the grid: Enabling scalable virtual organizations. *Int'l J. Supercomputer Applications, 15*(3).

Glite. (n.d.). *Glite middleware*. Retrieved from http://glite.web.cern.ch/glite

Globus. (n.d.). *Globus alliance toolkit homepage*. Retrieved from http://www.globus.org/toolkit/

Herveg, J. (2006). *The ban on processing medical data in European law: Consent and alternative solutions to legitimate processing of medical data in healthgrid. Proc. Healthgrid* (*Vol. 120*, pp. 107–116). Amsterdam, The Netherlands: IOS Press.

JSDL. (n.d.). *Job submission description language (jsdl) specification, v.1.0*. Retrieved from http://www.gridforum.org/documents/GFD.56.pdf

Kalra, D., Singleton, P., Ingram, D., Milan, J., MacKay, J., Detmer, D., & Rector, A. (2005). Security and confidentiality approach for the clinical e-science framework (clef). *Methods of Information in Medicine*, *44*(2), 193–197.

LHC. (n.d.). *LHC computing grid project*. Retrieved from http://lcg.web.cern.ch/LCG

Malin, B. (2002). *Compromising privacy with trail re-identification: The Reidit algorithms*. (CMU Technical Report, CMU-CALD-02-108), Pittsburgh.

Montagnat, J., Frohner, A., Jouvenot, D., Pera, C., Kunszt, P., & Koblitz, B. (2007). A secure grid medical data manager interfaced to the glite middleware. *Journal of Grid Computing*, *6*(1).

NIST. (2007). *Special publication 800-88: Guidelines for media sanitization by the national institute of standards and technology*. Retrieved from http://csrc.nist.gov/publications/nistpubs/#sp800-88

Olabarriaga, S. D., Nederveen, A. J., Snel, J. G., & Belleman, R. G. (2006). *Towards a virtual laboratory for FMRI data management and analysis. Proc. HealthGrid 2006* (*Vol. 120*, pp. 43–54). Amsterdam, The Netherlands: IOS Press.

Scarfone, K., & Mell, P. (2009) An analysis of CVSS version 2 vulnerability scoring. *Proceedings of the 3rd. Int'l Symposium on Empirical Software Engineering and Measurement (ESEM'09),* (pp. 516-525).

Sweeney, L. (2002). K-anonymity: A model for protecting privacy. *International Journal of Uncertainty*. *Fuzziness and Knowledge-Based Systems*, *10*(5), 557–570. doi:10.1142/S0218488502001648

Travostino, F., Daspit, P., Gommans, L., Jog, C., de Laat, C. T. A. M., & Mambretti, J. (2006). *S*eamless live migration of virtual machines over the man/wan. *Future Generation Computer Systems*, *22*(8), 901–907. doi:10.1016/j.future.2006.03.007

U.S. Congress (1996). *Health insurance portability and accountability act,* 1996.

U.S. Safe Harbor Framework. (n.d.). Retrieved from http://www.export.gov/safeharbor/

Van 't Noordende, G., Balogh, A., Hofman, R., Brazier, F. M. T., & Tanenbaum, A. S. (2007). *A secure jailing system for confining untrusted applications*. 2nd Int'l Conf. on Security and Cryptography (SECRYPT), (pp. 414-423). Barcelona, Spain.

Van 't Noordende, G. J., Brazier, F. M. T., & Tanenbaum, A. S. (2004). *Security in a mobile agent system*. 1st IEEE Symp. on Multi-Agent Security and Survivability, Philadelphia.

WGBO. (1994). *Dutch ministry of health, welfare and sport – WGBO*. Retrieved from http://www.hulpgids.nl/wetten/wgbo.htm

WMO. (1998). *Dutch ministry of health, welfare and sport - WMO*. Retrieved from http://www.healthlaw.nl/wmo.html.

Xu, L. (2005). *Hydra: A platform for survivable and secure data storage systems*. ACM StorageSS.

ENDNOTE

[1] Note that if any (untrusted) CRB could query the TSRB directly for the locations from which data is available, the result can reveal whether a given data file exists or not. Such information may be considered sensitive in itself, as outlined earlier.