Providing security to the Desktop Data Grid

Jesus Luna[*] Michail Flouris[†] Manolis Marazakis
Angelos Bilas[‡]
Institute of Computer Science
Foundation for Research and Technology - Hellas (FORTH)
PO Box 1385. GR-71110. Heraklion, Greece.
{jluna, flouris, maraz, bilas}@ics.forth.gr

Abstract

Volunteer Computing is becoming a new paradigm not only for the Computational Grid, but also for institutions using production-level Data Grids because of the enormous storage potential that may be achieved at a low cost by using commodity hardware within their own computing premises. However, this novel "Desktop Data Grid" depends on a set of widely distributed and untrusted *storage nodes, therefore offering no guarantees about neither availability nor protection to the stored data. These security challenges must be carefully managed before fully deploying Desktop Data Grids in sensitive environments (such as eHealth) to cope with a broad range of storage needs, including backup and caching.*

In this paper we propose a cryptographic protocol able to fulfil the storage security requirements related with a generic Desktop Data Grid scenario, which were identified after applying an analysis framework extended from our previous research on the Data Grid's storage services. The proposed protocol uses three basic mechanisms to accomplish its goal: (a) *symmetric cryptography and hashing,* (b) *an Information Dispersal Algorithm and the novel* (c) *"Quality of Security"* (QoSec) *quantitative metric. Although the focus of this work is the associated protocol, we also present an early evaluation using an analytical model. Our results show a strong relationship between the assurance of the data at rest, the QoSec of the Volunteer Storage Client and the number of fragments required to rebuild the original file.*

[*]This work was carried out for the CoreGRID IST project n°004265, funded by the European Commission.

[†]M. Flouris is also with the Dept. of Computer Science, University of Toronto, Toronto, Ontario M5S 3G4, Canada.

[‡]A. Bilas is also with the Dept. of Computer Science, University of Crete, P.O. Box 2208, Heraklion, GR 71409, Greece.

1 Introduction

The Desktop Data Grid is a specific type of distributed system, where shared resources (processor or storage) are provided in a volunteer fashion by the participants. These environments potentially provide commodity resources not only for CPU-intensive tasks, but also for applications that require significant amounts of memory, disk space and network throughput. For example, the widely used BOINC infrastructure [2] provides for a typical volunteer computing project a sustained rate of 95.5 TFLOPS, 7.74 Petabytes of storage and an access rate of 5.27 Terabytes per second [10]. However the potential computing power of volunteer systems is even greater: The number of Internet connected PCs is projected to reach 1 billion by 2015 [9], which means several PetaFLOPS of computing power and a storage capacity (around one Exabyte) able to exceed the one provided by any centralized system. Therefore, it is not surprising that nowadays a lot of ongoing effort is targeting Volunteer Computing as a new paradigm for both the Computational and the Data Grid. A representative example is the Lattice Project [11], which is expanding the reach of Grid computing by creating a system that combines the Globus Toolkit [17] and BOINC.

Tapping unused PC capacity promises efficiency and reliability comparable to traditional storage attached networks (SANs) minus the associated costs and management headaches that come with large-scale SAN deployments. This emerging approach is quite attractive for small to midsize businesses that would like to store large amounts of data in their premises, or even to backup critical data [7]. Even institutions already using production-level Data Grid infrastructures (such as EGEE [6]) may find useful the Desktop Data Grid paradigm, as a cache for large-size files within their own LANs. However, for the successful deployment of the Volunteer Computing paradigm into any data storage-related scenario (this particular application field will be referred to as the *Desktop Data Grid* throughout this paper), the *security* aspect arises as a critical parameter.

Figure 1 shows the basic elements and interactions that usually take place in a Desktop Data Grid. Grid Users interact with the Desktop Grid Service, either generating or using information (data producers and/or consumers). Also there is a set of security-related services grouped under the generic term *Door node*, which are the servers performing all the authentication and authorization processes for involved entities (users and resources). On the other hand, the Project Server (e.g a BOINC server) manages the metadata and the data itself prior to its stage to the pool of Volunteer Storage Clients (VSCs) which provide storage for the Desktop Data Grid. In most volunteer installations, all communication is initiated by the VSCs, in order to facilitate firewall and NAT traversal.

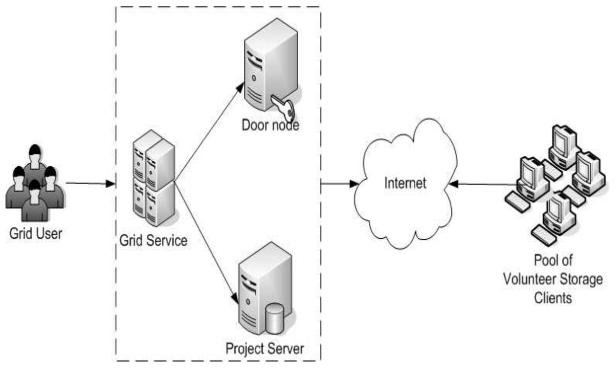

Figure 1. A Typical Desktop Data Grid.

As a preliminary step in designing the security mechanisms for a generic Desktop Data Grid, the first part of this paper extends the analysis framework that we applied in [21] to identify its particular security issues and challenges. The second part of this work relies on the requirements obtained via the security analysis to propose a cryptographic protocol able to protect the data both on the wire and at rest (in VSCs), applying three basic mechanisms *(i)* cryptography, *(ii)* data fragmentation, and *(iii)* a novel quantitative security evaluation metric.

The rest of this paper is organized as follows. Section 2 analyzes the security issues of the typical Desktop Data Grid by applying an extended framework used in our previous research about Data Grid's Storage Services. Section 3 introduces the proposed security protocol and analyzes the security guarantees it provides to Desktop Data Grids, while section 4 shows how data assurance can be improved by using the proposed mechanisms. Section 5 surveys the state-of-the-art related with security in Desktop Data Grids and associated technologies. Finally, section 6 summarizes our conclusions and outlines future work.

2 Security Analysis

From the point of view of the Desktop Data Grid being studied, its subsystems may become attacked in several ways; for the purposes of our research, the framework proposed by [26] and extended in [21] will be used to find out the main concerns related to the security of its data and metadata. In a glimpse, the use of this framework consists of determining the basic components related with the system's security (players, attacks, security primitives, granularity of protection, and user inconvenience), so that afterwards they can be summarized to clearly represent the security requirements. Using these concepts, the rest of this section analyzes the Desktop Data Grid shown in figure 1.

2.1 Security Framework components

The first step in our analysis is to identify the elements that play a security-related role in a generic Desktop Data Grid:

1. *Players:* Three data readers/writers are involved *(i)* the Volunteer Storage Client; *(ii)* the Project Server, which also implements a metadata service that gathers information about stored files; and *(iii)* the WAN links (referenced also as the "wire") conveying information between VSCs and Project Servers. For this framework the Grid Service and the Grid user are not considered as players.

2. *Attacks:* The generic attacks that may be executed over the Desktop Data Grid are related with *(i)* Adversaries on the wire; *(ii)* Adversaries on the infrastructure servers (Project Server or door node); *(iii)* Revoked users on the door node; and *(iv)* Adversaries with *full control* of the site services (Volunteer Storage Clients). Each one of these attacks may result in data being leaked, changed or even destroyed.

3. *Security Primitives:* Two security operations take place into a typical Desktop Data Grid: *(i)* when Grid users are authenticated by door nodes before accessing a Grid Service; and *(ii)* when the door node also authorizes Grid users.

4. *Trust Assumptions:* We have considered that in general *(i)* VSCs have full control over the data stored on them (which means that a malicious VSC may leak, change or destroy it); *(ii)* data are encrypted on the wire using a secure channel (i.e. SSL); *(iii)* data are unprotected on the VSCs' disks, but protected at the Project Server; *(iv)* infrastructure servers are trusted; and finally *(v)* the VSC's client software is trusted only when coming from a Project Server (i.e. digitally signed code distribution).

5. *User inconvenience:* Desktop Data Grids only require the installation of a client-side software at the volunteer nodes, and these nodes are only expected to provide storage space, therefore keeping processor overhead to a minimum.

2.2 Results of the security analysis

Using the elements identified in section 2.1 and after applying the analysis framework from [21] we have obtained a set of security issues which have been summarized in table 1. Results are categorized by possible attacks (main columns) and types of damage – the Leak, Change, Destroy sub-columns –. Cells marked with a "Y" mean that the system (row) is vulnerable to the type of damage caused by this particular attack. Cells marked with a "N" mean that the attacks are not feasible or may not cause a critical damage. If a particular attack does not apply at all to the Desktop Data Grid, then the cell is marked with a "-".

Table 1. Summary of security issues related with Desktop Data Grids

Attack	*Adversary on wire*			*Adversary on infs servers*			*Revoked user on door node*			*Adversary w/site services*		
Damage	L	C	D	L	C	D	L	C	D	L	C	D
Desktop Data Grid	N	N	Y	N	N	N	Y	Y	Y	Y	Y	Y

Our conclusion from the previous security analysis is that there are two main security concerns in a Desktop Data Grid. The first issue is related to the Infrastructure Servers, in particular when revoked users have colluded with door nodes. We believe, however, that it is feasible to extrapolate the proposed mechanisms based on previous research on validation for Computational Grid's users [22]; therefore these attacks do not generally represent a major security risk. On the other hand, we also identified a second group of critical vulnerabilities related with the VSCs, which are considered *untrusted* and *heterogeneous* (different hardware and software configurations), thus offering a non-uniform level of protection to the stored information. Consequently, the rest of this paper is focused on a security protocol designed to protect a Desktop Data Grid environment from adversaries colluded with untrusted Volunteer Nodes.

3 Security Protocol

A security mechanism conveying sufficient data protection for the Desktop Data Grid should provide a fair balance between performance overheads and data confidentiality, integrity and availability. With these requirements, we designed a security protocol that achieves its goals based on three techniques. The first is an application of symmetric cryptography, whereby, before staging data into the VSCs, the Project Server encrypts the data file with a *symmetric master key* composed of the file's hash appended to a nonce (i.e. a timestamp). With this technique, we provide confidentiality (as the VSCs are unaware of the encryption key) and integrity (via the hash function). As an optional authorization feature, when a Grid Client is retrieving a file, we propose the use of a public-key cryptosystem for encrypting the symmetric master key, thus ensuring that only the legitimate owner of the corresponding private key is able to decrypt the symmetric master key and therefore the requested file.

A second technique, the Information Dispersal Algorithm, provides *high availability and assurance* for the Desktop Data Grid by means of *data fragmentation*. In a fragmentation scheme [25], a file *f* is split into *n* fragments, all of these are signed and distributed to *n* remote servers, one fragment per server. The user then can reconstruct *f* by accessing *m* arbitrarily chosen fragments ($m \leq n$). The fragmentation mechanism can also be used for storing long-lived data with high assurance: *f* is fragmented in just a few pieces (small *m*) and dispersed to a large number of nodes (large *n*) (a replication-like approach). When fragmentation is used along with encryption, data security is enhanced: An adversary colluded with the VSCs has to compromise and retrieve *m* fragments of file *f*, and then has to break the encryption system *K*. For the proposed protocol, the dispersal algorithm is based on the Reed-Solomon *erasure* code [24]. This means that when a device fails, shuts down or is compromised, then the system recognizes this event, as opposed to an *error*, where a device failure is manifested by storing and retrieving incorrect values, that can only be detected using codes that are embedded within the data. We selected the Reed-Solomon erasure code for our protocol taking into consideration its successful applications in several fault-tolerant systems (see also section 5).

Finally, a third security technique takes into account the heterogeneity of the Desktop Data Grid's nodes. Our premise is that if subsets of VSCs with analogous features are clearly identified and their security level quantified, then any user of the Desktop Data Grid is able to request storage space *only* at nodes fulfilling some minimum guarantees or *Quality of Security (QoSec)* level. To this end, we propose a policy-based approach, where *(i)* each VSC is associated with a Policy modeling its security behavior and, *(ii)* an evaluation methodology (presented in section 3.1) for computing a quantitative QoSec metric associated with such policy. Previous research [14] used the Certificate Policy [16] associated with a Grid user's X.509 certificate to perform the REM evaluation. We believe that an analogous approach for the Volunteer nodes is quite feasible. This approach does not add considerable overhead to the protocol, mainly because the evaluation can be performed by the Project Server at any time (not necessarily in real-time) as soon as the VSC's policy has been retrieved. Once evaluated, a numeric QoSec factor can be associated with every VSC; moreover, a proprietary set of reference levels can be established to group these discrete values into a major set, for example (L_0, L_1, L_2, L_3) which represents four different QoSec. Any Grid Client storing data will request a minimum QoSec to be fulfilled by the VSCs. An important challenge with this proposal is the definition of VSC's security policy itself. Ongoing research in the context of the CoreGRID NoE [5] is moving towards defining the security policy associated with a generic storage element, and as members of this group we plan to implement such a policy on the Desktop Data Grid. A second challenge is the VSC security policy's *audit process*, because the proposed evaluation would be pointless if the storage node is not honoring its policy. However, there are some "controlled" scenarios where this *audit* can be performed with confidence; for example, a Desktop Data Grid overlapping different institutions that are members of the same *Grid Policy Management Authority* (PMA), that is, clients using in their installations digital certificates issued by a Certification Authority that has passed a quite strict accreditation process, enabling it to inter-operate in Grid projects only with institutions members of the same PMA.

The following subsection presents important details of the proposed evaluation mechanism, called the Reference Evaluation Methodology (REM). Subsections 3.2 and 3.3 then describe the interactions taking place when a Grid client writes and reads a file using the proposed protocol. When presenting the proposed protocol it must be taken into account that *(i)* all communication taking place among the involved parties is conveyed through encrypted channels (i.e. via SSL tunnels) and, *(ii)* communication is initiated by the VSCs. Finally, file deletion is unaffected by our protocol and follows the same principles from [2], where the Project Server creates an specific message for this operation (after authenticating and authorizing the requestor).

3.1 The Reference Evaluation Methodology in a glimpse

The core of our proposal to quantify the Quality of Security associated with a Volunteer node is the REM methodology introduced in [15], which has been successfully applied in other security-related environments, i.e. Public Key Infrastructures (PKI). Basically the REM takes a policy to evaluate, formalize it to ease the further evaluation step over an homogeneous metric space, uses a set of reference levels to apply a distance criteria and finally obtains a number that corresponds to the policy's security level. The rest of this section presents an illustrative example that shows basic use of the REM methodology to evaluate a single security provision for a VSC. To evaluate the full security policy, this same procedure should be applied to each one of its provisions. Interested readers that would like to take a closer look at the details and formalisms behind REM should refer to [13].

Step 1: Policy formalization. In REM, a security policy is formed via a set of individual security provisions, so let us suppose that a VSC's security policy contains the provision called $P_x =$"User log-on mechanism". To formalize a provision it is necessary to state the possible values that are allowed for it to take into the VO. For our example let us suppose the following values for P_x (from least to most secure) "None, Password, Smartcard, Biometric", this means that P_x has a cardinality of 4.

Step 2: The evaluation technique. If for a particular VSC the user has to login via Smartcard, then the provision P_x takes as value the *vector* $(1, 1, 1, 0)$, which means that it is more secure than the Password-only mechanism (vector $(1, 1, 0, 0)$), but less than Biometry $(1, 1, 1, 1)$. Using REM's nomenclature, this implies a *Local Security Level* or $LSL_x = 3$. It is easy to see that after evaluating a whole formalized security policy for a particular VSC, we will have a set of vectors or in other words, a *matrix* M_x.

To obtain the *Global Security Level* (*QoSec*) for this particular VSC, it is necessary to have a new zero-matrix called M_ϕ which contains only *0's* as its elements. Notice that M_ϕ represents the least secure VSC that can be found into a Desktop Data Grid installation. The QoSec in our example is defined as the Euclidean distance between M_x and M_ϕ, that is:

$$d(M_x, M_\phi) = \sqrt{\sigma(M_x - M_\phi, M_x - M_\phi)}. \qquad (1)$$

Where:

$$\sigma(M_x - M_\phi, M_x - M_\phi) = Tr\left((M_x - M_\phi)(M_x - M_\phi)^T\right). \qquad (2)$$

Obviously the REM technique considers more complex scenarios, where for example different provisions have different security weights (some provisions are more important than others) and even different cardinalities [13].

3.2 Grid Client writing a file

Any authenticated and authorized Grid Client must perform the following interactions when writing a file to the Desktop Data Grid with the proposed protocol:

1. The Producer *P*, generates the clear-text data f and sends it to the Project Server.
2. On the Project Server, the cryptographic hash *H(f)* is computed, concatenated with the nonce *T* and used as the symmetric key to encrypt the file *f*. That is:

$$E_{H(f)|T}(f). \tag{3}$$

   This *master encryption key* is stored as part of the associated metadata into the Project Server. Also a numeric $QoSec_f$, representing the Security Level requested by the Producer, is associated with the data *f*.

3. The encrypted data from (3) is fragmented with the IDA and the Project Server stages it for downloading only to those Volunteer Storage Clients fulfilling the minimum requested $QoSec_f$. Notice that the Project Server has previously computed offline the QoSec associated with each participating VSC.
4. The VSCs queries the Project Server and if required to do so will download and store the encrypted fragments.

3.3 Grid Client reading a file

When an authenticated and authorized Grid Client requests a file from the Desktop Data Grid, the following protocol is performed:

1. The Project Server requests from the VSCs to upload the fragments required to rebuild the file solicited by the user.
2. The VSCs query the Project Server and if required to do so, will upload the encrypted fragments.
3. The Project Server defragments and decrypts with *H(f)*$|$*T* the uploaded data from (3) to rebuild the file *f*.
4. *Optional:* If required, instead of directly decrypting (3), the Project Server may re-enforce authorization by encrypting the stored master key *H(f)*$|$*T* with the public key of the Consumer (Pub_C), that is:

$$E_{Pub_C}(H(f)|T). \tag{4}$$

5. The data consumer retrieves the clear-text file f or, if still encrypted (previous step), then *(i)* with its private key obtains *H(f)*$|$*T* from (4) and, *(ii)* uses this master key to obtain the clear-text file (*f*) from (3), also verifying *f*'s integrity and freshness.

The following section presents via an analytical model the strong relationship between the security of the data at rest, the QoSec of the Volunteer Storage Client and the number of fragments required to rebuild the original file.

4 Results: QoSec and Data Assurance for Production Grids

We use the model presented in [23] to evaluate the *assurance* of a file stored in a Desktop Data Grid according to the proposed protocol. Assurance is defined as the probability that the cited file has not been compromised under the assumption that the system was the target of a successful attack. Formally speaking, we assume that a Desktop Data Grid with *N*-Volunteer Storage Clients is built in such a way that the same configuration (and therefore the same *security level*) is present in at most $\lceil \lambda N \rceil$ volunteer nodes, where $\lambda \in \mathcal{R}$ and $0 < \lambda < 1$. A potential attacker could then compromise at most $\lceil \lambda N \rceil$ servers, because the same vulnerabilities are to be found in all of them. Therefore, it makes sense to state that a Desktop Data Grid can be considered *secure* if it is composed of heterogeneous volunteer computers, where of course $\lambda \approx 0$. According to the authors of [23], this latter fact can be represented by the conditional probability of the event "at most *m-1* VSCs, each storing a fragment of *f*, are down or compromised" given the event "the Desktop Data Grid has been attacked"; this probability is known as the *distribution assurance* $A_\phi(\mu)$ and for a dispersal algorithm μ applied over a file f will be denoted by (5).

$$A_\phi(\mu) = 1 - \sum_{i=m}^{n} \frac{\binom{\lceil \lambda N \rceil}{i}\binom{\lfloor N - \lambda N \rfloor}{n-i}}{\binom{N}{n}}. \tag{5}$$

According to the analytical model (5), it is more likely to find *subsets* of $\lceil \lambda N \rceil$-Volunteer Nodes sharing the same configuration. Thus, a better data assurance is obtained with highly secure ($QoSec \to \infty$) *or* heterogeneous ($\lambda \to 0$) volunteer nodes. Therefore:

$$QoSec \propto \frac{1}{\lambda} \Rightarrow QoSec = \frac{k}{\lambda}. \tag{6}$$

We can rewrite (5) in terms of the proposed *QoSec* with the factor $k = 1$:

$$A_\phi(\mu) = 1 - \sum_{i=m}^{n} \frac{\binom{\lceil \frac{N}{QoSec} \rceil}{i} \binom{\lfloor N - \frac{N}{QoSec} \rfloor}{n-i}}{\binom{N}{n}}. \quad (7)$$

Next, we examine a hypothetical (but feasible) Desktop Data Grid installed over VSCs from three different institutions: Greece's HellasGrid, CERN's LCG, and Spain's IRISGrid. Currently each of these installations is a member of the same Grid Policy Management Authority, which means that they fulfill a minimum set of security rules (provisions) established -and audited- by the European Grid PMA (EUGridPMA) according to its "Authentication Profile" document [1]. Our belief is that these three installations may provide an initial Virtual Organization suitable for a production-level Desktop Data Grid (we will elaborate more on this idea in section 6). As a first step, we applied the REM technique to compute the numeric QoSec associated with the Certificate Policy [19] subsection called "Technical Security Controls" -mostly related with the Volunteer Nodes of this hypothetical Desktop Data Grid- of HellasGrid (QoSec=4.47), CERN (QoSec=6.00), IRISGrid (QoSec=5.48) and the minimum reference EUGridPMA (QoSec=4.24).

The relationship between the assurance for the data at rest and the computed QoSec can be obtained by plotting expression (7), just as seen in figure 2. For the evaluation, we consider a Grid with 100 VSCs (*N=100*), implementing file fragmentation into 15 pieces (*n=15*). The figure shows that, despite the small difference between the computed QoSecs, CERN's nodes achieve the greatest data assurance (≈ 1) with the smallest number of fragments. On the other hand, a Grid installation fulfilling only the minimum security requirements (QoSec=4.24) requires the retrieval of more fragments from the volunteer nodes in order to fully defragment the stored file with the same data assurance.

The results presented in this section are a first approach to compute the Storage Element's QoSec using a subset of provisions from a Certification Policy; however, in a real-world installation we should evaluate a comprehensive Security Policy specifically built for this resource. Section 6 will return to this topic while outlining our on-going research.

5 State of the Art

As far as we know the *Storage@home* system [12] is the only proposal to build a Desktop Data Grid with a security mechanism analogous to the one presented in this paper. In *Storage@home*, each file is encrypted and split into four

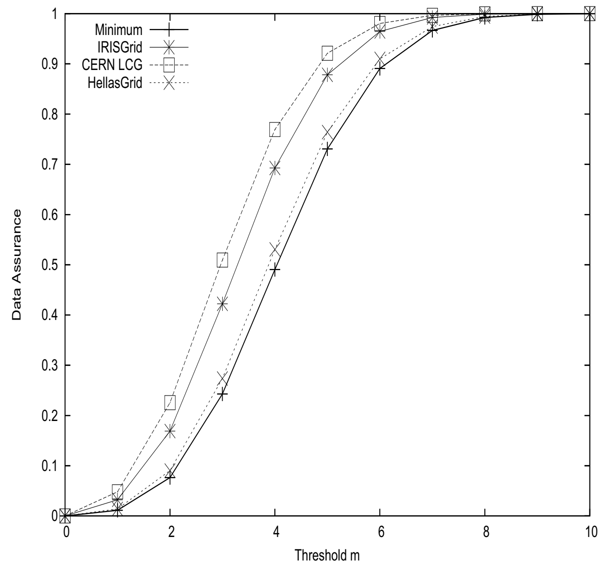

Figure 2. Relationship between data assurance and QoSec. The horizontal axis represents the number of fragments required to retrieve from the VSCs to rebuild the original file.

copies striped across ten hosts. Despite the fact no more details are given about the security subsystem (i.e., where encryption is performed?), it seems that all volunteer nodes are treated equally by the so-called Policy Engine, and instead of fragmentation their system uses a pure-replication scheme (therefore achieving a small data assurance). However for the reasons explained in section 3 we can foresee an important security enhancement if the QoSec proposed in this paper is used for distributing data within this system (maybe using the "reward" subsystem already implemented by the authors).

Security in widely used volunteer computing systems like BOINC [3] cope with problems like the ones identified by our analysis using probabilistic techniques to detect result falsification (data change attack). However contrary to our proposal, no protection is provided against theft of project files (both input and output files are not encrypted) under the assumption that anyway data resides in cleartext in memory, where it is easy to access with a debugger. For a Desktop Data Grid this is unacceptable, because only the authenticated and authorized Grid user should have access to this information.

Despite it is not a Desktop Data Grid, POTSHARDS [27] implements an storage system for long-time archiving that does not use encryption, but a mechanism called "probably secure secret splitting" that fragments the file to store prior to distributing it across separately-managed archives. In general their approach is interesting to cope with the management problems posed by cryptosystems and long-living data, however the security achieved only by fragmenting the files could not be enough for some highly-sensitive environments, just as seen in this paper.

The Farsite system [8] provides security and high availability by storing encrypted replicas of each file on multiple machines. However the use of replication instead of fragmentation reduces data assurance (the attacker just needs to compromise one node to retrieve a full file) and may not be bandwidth-wise for big files.

A proposal that copes with the problem of untrusted storage servers is called OceanStore [20], where stored data are protected with redundancy (also using erasure codes) and cryptographic mechanisms. An interesting feature in OceanStore is the ability to perform server-side operations directly on the encrypted data, this increases system's performance without sacrificing security. On the other hand, contrary to our proposal, the authors still consider that all the storage nodes are homogeneous and therefore offer the same QoSec.

Finally is worth to mention Cleversafe [4], which implements also an Information Dispersal Algorithm (based on the Reed-Solomon algorithm) for its open-source *Dispersed Storage Project*. Despite this system does not use encryption at all, the authors provide a flexible API that we may extend in a near future -for the purposes of this research- with the proposed cryptography and QoSec mechanisms.

6 Conclusions

Providing security to the Desktop Data Grid is critical for its deployment in production environments, where its full potential can be developed in terms of storage (i.e. for backup and caching) and computing power. However, it is mandatory to establish guarantees over the security provided to the data and metadata being managed in these systems in order to avoid unnecessary risks. Towards this goal, we presented a proposal for enhancing the overall security of the Desktop Data Grid, with a protocol that makes use of three basic techniques (cryptography, data fragmentation and a quantitative security metric) to cope with untrusted Volunteer nodes participating in these environments. The proposed protocol was focused on vulnerabilities indicated by applying an extended security analysis framework, developed in our previous research for the Data Grid storage services.

Even though the use of cryptography and Information Dispersal Algorithms is not new for securing data and metadata in distributed systems, the main contribution of our research consists of enhancing these mechanisms with the adoption of a numeric *Quality of Security -QoSec-* level representing the guarantees offered by the Volunteer Storage Client to the Grid User's stored data. Applying an analytical model we have shown for three production Grids the strong relationship between the data assurance, the number of fragments required to retrieve a file, and the volunteer node's QoSec. We strongly believe in the security advantages of using the QoSec factor as an input to the Information Dispersal Algorithm; however, the following question remains open: how far are we from obtaining a similar result in a more generalized and open environment? Our future work goes precisely in the direction of defining a policy for storage elements, beyond the Certification Policy used in this paper, able to model with confidence their security properties and the Grid user's expectation from the authentication, authorization, confidentiality, integrity, privacy and availability point of view. This work is currently taking place in the content of the CoreGrid NoE [5].

On the other hand, we have to consider an important limitation inherent to Desktop Data Grids: the VSC's bandwidth. To improve computational performance over the stored data, future research will consider executing code at the VSC directly (contrary to moving the data itself). An important security challenge here are the fragmented and encrypted data, because for the computation to take place it might be necessary to retrieve fragments from other VSCs and, of course, decrypt them afterwards (which means disclosing the corresponding encryption key to the VSC). A first approach could be to do this only on those VSCs with a "high QoSec" (e.g. those using cryptographic hardware).

Despite the usefulness of the QoSec concept, it is important to avoid abuses from Grid Clients (such as a client requesting always a high QoSec even for non-critical data). To this end, future research should provide the Project Server with a *QoSec validation* mechanism, so that QoSecs are also assigned to Grid Clients or to the data objects themselves, to decide the maximum security level that should be requested from VSCs. This process can be seen as "QoSec-based authorization".

A final research line aims towards implementing the proposed mechanism within the BOINC system, so we become able to fine-tune its performance costs (mainly related with the cryptographic operations) to achieve a fair balance with the security required by production environments. A potential testbed for this protocol is the ICGrid project [18], currently being developed jointly by the University of Cyprus and the General Hospital of Nicosia. The motivation is to use some of the Hospital's idle storage resources to build a Desktop Data Grid, similar to the model described in this paper, not only for backing up large amounts of patient's data (possibly confidential), but also for providing a temporary cache for large files being retrieved from EGEE sites.

Acknowledgments

We thankfully acknowledge the support of the European FP6-IST program through the UNISIX project and the CoreGRID Network of Excellence. We also thank Marios D. Dikaiakos from the University of Cyprus for the useful discussions concerning a practical and real-life health-oriented Grid system.

References

[1] Classic ap profile version 4.03. http://www.eugridpma.org/igtf/IGTF-AP-classic-20050905-4-03.pdf, 2005.

[2] Berkeley open infrastructure for network computing. http://boinc.berkeley.edu/, 2007.

[3] Boinc: Security issues in volunteer computing. http://boinc.berkeley.edu/trac/wiki/SecurityIssues, 2007.

[4] Cleversafe. http://www.cleversafe.com, 2007.

[5] Coregrid network of excellence. http://www.coregrid.net, 2007.

[6] Egee - enabling grids for e-science. http://www.eu-egee.org/, 2007.

[7] Tapping pc resources for storage needs. http://www.internetnews.com/storage/article.php/3720931, 2007.

[8] A. Adya, W. J. Bolosky, M. Castro, G. Cermak, R. Chaiken, J. R. Douceur, J. Howell, J. R. Lorch, M. Theimer, and R. Wattenhofer. Farsite: Federated, available, and reliable storage for an incompletely trusted environment. In *OSDI*, 2002.

[9] D. P. Anderson. Boinc: A system for public-resource computing and storage. In R. Buyya, editor, *GRID*, pages 4–10. IEEE Computer Society, 2004.

[10] D. P. Anderson and G. Fedak. The computational and storage potential of volunteer computing. In *CCGRID*, pages 73–80. IEEE Computer Society, 2006.

[11] A. Bazinet and M. Cummings. The lattice project: a grid research and production environment combining multiple grid computation models. In Weber, M. H. W. (Ed.) Distributed and Grid Computing - Science Made Transparent for Everyone. Principles, Applications and Supporting Communities. Tectum. To appear.

[12] A. L. Beberg and V. S. Pande. Storage@home: Petascale distributed storage. In *IPDPS*, pages 1–6. IEEE, 2007.

[13] V. Casola, A. Mazzeo, N. Mazzocca, and V. Vittorini. A policy-based methodology for security evaluation: A security metric for public key infrastructures. *Journal of Computer Security*, 15(2):197–229, 2007.

[14] V. Casola, N. Mazzocca, J. Luna, O. Manso, and M. Medina. Static evaluation of certificate policies for grid pkis interoperability. In *ARES '07: Proceedings of the The Second International Conference on Availability, Reliability and Security*, pages 391–399, Washington, DC, USA, 2007. IEEE Computer Society.

[15] V. Casola, R. Preziosi, M. Rak, and L. Troiano. A reference model for security level evaluation: Policy and fuzzy techniques. *J. UCS*, 11(1):150–174, 2005.

[16] S. Chokhani, W. Ford, R. Sabett, C. Merrill, and S. Wu. Internet X.509 Public Key Infrastructure - Certificate Policy and Certification Practices Framework. RFC 3647 (Informational), 2003.

[17] I. T. Foster. The globus toolkit for grid computing. In *CCGRID*, page 2. IEEE Computer Society, 2001.

[18] K. Gjermundrod, M. Dikaiakos, D. Zeinalipour-Yazti, G. Panayi, and T. Kyprianou. Icgrid: Enabling intensive care medical research on the egee grid. In *From Genes to Personalized HealthCare: Grid Solutons for the Life Sciences. Proceedings of HealthGrid 2007*, pages 248–257. IOS Press, 2007.

[19] R. Housley, W. Polk, W. Ford, and D. Solo. Internet X.509 Public Key Infrastructure - Certificate and Certificate Revocation List (CRL) Profile. RFC 3280 (Informational), 2002.

[20] J. Kubiatowicz, D. Bindel, Y. Chen, S. E. Czerwinski, P. R. Eaton, D. Geels, R. Gummadi, S. C. Rhea, H. Weatherspoon, W. Weimer, C. Wells, and B. Y. Zhao. Oceanstore: An architecture for global-scale persistent storage. In *ASPLOS*, pages 190–201, 2000.

[21] J. Luna et al. An analysis of security services in grid storage systems. In *CoreGRID Workshop on Grid Middleware 2007*, June 2007.

[22] J. Luna, M. Medina, and O. Manso. Using ogro and certiver to improve ocsp validation for grids. In Y.-C. Chung and J. E. Moreira, editors, *GPC*, volume 3947 of *Lecture Notes in Computer Science*, pages 12–21. Springer, 2006.

[23] A. Mei, L. V. Mancini, and S. Jajodia. Secure dynamic fragment and replica allocation in large-scale distributed file systems. *IEEE Trans. Parallel Distrib. Syst.*, 14(9):885–896, 2003.

[24] J. S. Plank. A tutorial on reed-solomon coding for fault-tolerance in raid-like systems. Technical Report CS-96-332, University of Tennessee, Department of Computer Science, 1997.

[25] M. O. Rabin. Efficient dispersal of information for security, load balancing, and fault tolerance. *J. ACM*, 36(2):335–348, 1989.

[26] E. Riedel, M. Kallahalla, and R. Swaminathan. A framework for evaluating storage system security. In D. D. E. Long, editor, *FAST*, pages 15–30. USENIX, 2002.

[27] M. W. Storer, K. M. Greenan, E. L. Miller, and K. Voruganti. Secure, archival storage with potshards. In *FAST'07: Proceedings of the 5th conference on USENIX Conference on File and Storage Technologies*, pages 11–11, Berkeley, CA, USA, 2007. USENIX Association.