Computer Standards & Interfaces 32 (2010) 230–245

Contents lists available at ScienceDirect

Computer Standards & Interfaces

journal homepage: www.elsevier.com/locate/csi

A security framework for a workflow-based grid development platform

José L. Vivas, Carmen Fernández-Gago, Javier Lopez *, Andrés Benjumea

Department of Computer Science, Complejo Tecnológico, Campus de Teatinos, University of Malaga, 29071, Malaga, Spain

ARTICLE INFO

Available online 21 April 2009

Keywords:
Grid
Security framework
Security services
Virtual organizations

ABSTRACT

This paper describes the security framework that is to be developed for the generic grid platform created for the project GREDIA. This platform is composed of several components that need to be secured. The platform uses the OGSA standards, so that the security framework will follow GSI, the portion of Globus that implements security. Thus, we will show the security features that GSI already provides and we will outline which others need to be created or enhanced.

1. Introduction

A grid may be defined as a collection of computing resources distributed over a local or wide area network, and available to an end user as a single large computing system. Originally, the grid focused on the areas of computing power, data access, and storage resources. It was intended for large-scale and distributed scientific computing that required efficient and dynamically determined access to large amounts of data and computational resources that are distributed along several independently administered networks. However, the use of grid computing has been expanding lately to include deployment of grid technologies within the context of business [38], which significantly widens the range of applicability of grid technologies. Standard interfaces for business services have also been leveraged by grid computing. Grid computing has been targeting such differing areas as finance, medicine, decision-making, collaborative design, and utility computing. The focus today is on coordinated resource sharing distributed across virtual organizations. However, shareable on-demand resources in commercial applications greatly complicate resource sharing and introduce new challenges related to federated security and integration.

Security has been a central issue in Grid computing from the outset, and has been regarded as the most significant challenge for grid computing [42]. This is particularly true for Enterprise Grids. Significant compromises in security might be the result of an inadequate understanding of the security implications of a grid. The security requirements and policies are determined largely by the architectures developed for these types of applications, which are distinguished from client–server architectures by the fact that grid environments assume a dynamic and simultaneous use of a large number of resources from a number of administrative domains. Although the intention has been from the outset to use available security mechanisms as much as possible, this requirement could not be met by mechanisms that were devised largely for insulating and protecting networks from their environment, as in intranets and virtual private networks. As a result, novel security technologies have been evolving all the time within the grid community, including solutions for the management of credentials and policies, new resource management protocols for co-allocation of multiple resources and for secure remote access to data and computing resources and new information query protocols and data management services [71].

In this paper we will focus on the development of a security framework for a generic grid development platform developed within the European project GREDIA, "Grid Enabled to Rich Media Content" [18]. The main objective of the project is to create a generic grid platform that will combine new and existing middleware to support secure business applications. GREDIA addresses both desktop and mobile terminals, being the latter one of the novelties of the project. In order to validate the platform two pilot applications will be used: news and banking applications.

The paper is organized as follows. Section 2 gives an overview on the Globus Toolkit. Section 3 outlines what are the security services to be taken into account in grids, paying special attention to authorization. Section 4 presents all the components of the GREDIA architecture and Section 5 how they are integrated into a single platform. Section 6 concludes the paper and outlines the future work.

2. Grid security architectures and infrastructures

2.1. OGSA security

Open Grid Services Architecture (OGSA) is a *service-oriented architecture* (SOA) that represents an evolution towards a grid system

* Corresponding author.
E-mail addresses: jlvivas@lcc.uma.es (J.L. Vivas), mcgago@lcc.uma.es (C. Fernández-Gago), jlm@lcc.uma.es (J. Lopez), abenjumea@lcc.uma.es (A. Benjumea).

0920-5489/$ – see front matter © 2009 Elsevier B.V. All rights reserved.
doi:10.1016/j.csi.2009.04.001

architecture based on Web services concepts and technologies, autonomic computing principles and open standards for integration and inter-operability. The first implementation of OGSA mechanisms corresponds to version 3 of the Globus toolkit, GT3.

OGSA builds on concepts and technologies from both the grid and Web services. It was created in order to meet the challenges related to the integration of services across virtual organizations (VOs) running on top of different native platforms [28].

Security arises at various levels of the OGSA architecture. The OGSA security model stipulates that security mechanisms should be pluggable and discoverable by service requesters from a service description, enabling service providers to select their preferred mechanisms. The Global Grid Forum's OGSA 1.0 [57] document targets security requirements including authentication and authorization, security infrastructures, perimeter security solutions, isolation, delegation, policy exchange, intrusion detection and protection, and secure logging. It also specifies security services associated with message integrity, confidentiality and privacy, auditing, intrusion prevention and access control.

2.2. The Globus Toolkit

The Global Grid Forum defines the Globus Toolkit (GT) as an "ecosystem" of components [32]. It contains a collection of components that have been proven to be useful and were selected from solutions that had been used in different grid applications. These solutions showed a potential for reusing because of their usefulness and generality. These solutions were then generalized in order to become useful for a wide variety of applications.

The Globus Toolkit is thus a collection of standard building blocks and tools that does not give a complete solution for any grid project, and very few of the GT components include user interface elements that are immediately useful to the application. GT provides solutions to the most common problems and promotes standard solutions. A grid application typically must assemble together a subset of these components, components from other sources, either off-the-shelf or specially tailored for the application, and application-specific code. These components must be organized and integrated by architects and system integrators according to the requirements of the specific application. Standard mechanisms used in the GT include SSL/TLS, LDAP v3, X.509 Proxy Certificates, SOAP, HTTP, and GridFTP. Reference implementations of new and proposed standards may also be provided, e.g. WSRF, DAI, WS-Agreement, WSDL 2.0, WSDM, SAML, and XACML.

The Globus Toolkit has evolved from a first version, GT1, to the most recent one, GT4.

- GT1 is tantamount to the Information Wide Area Year (I-WAY) project [26] conceived in 1995 with the idea to integrate high bandwidth networks.
- GT2 was released in 2001 and became a de facto standard for grids. In this version there was no common framework, the protocols were non-standard and the services were predefined, making it hard to develop new services.
- GT3 was based on the Open Grid Services Architecture (OGSA) specifications and used XML and protocol standards such as SOAP and WSDL to provide Web services interfaces [28]. Grid services are the core of GT3 and its corresponding infrastructure is called the Open Grid Services Infrastructure (OGSI).
- GT4 was launched in 2005. It featured a new implementation of the Web Services Resource Framework (WSRF) [86] and the Web Services Notification (WSN) [85] standards, thus rendering OGSI obsolete. WSRF can be viewed as a re-factoring of OGSI, which was considered long and complex, too object oriented and not fully integrated with Web services. WSRF is intended to be integrated into the family of Web service standards, allowing OGSA to be based directly on Web services. WSRF defines conventions within the context of current Web service standards for managing state so that applications may discover, inspect, and interact with stateful resources in standard and inter-operable ways.

2.2.1. The Grid Security Infrastructure (GSI)

The Grid Security Infrastructure (GSI) is the portion of the Globus Toolkit that implements security functionality. Its security model constitutes a de facto standard for grid security. GSI maps to local security mechanisms and gateways are used in order to translate from the common GSI infrastructure into the local site mechanisms. It supports the following features.

- *A public-key system*. GSI is based on public-key cryptography and thus supports privacy, integrity, and authentication. Both grid users and grid services need to have a X.509 certificate and a private key. User certificates can be managed in several ways. Certificates are issued by Certification Authorities that both users and services must trust.
- *Mutual authentication through digital certificates*. GSI supports three authentication methods: (i) X.509 certificates; (ii) username and password; (iii) anonymous authentication. GSI uses X.509 certificates to guarantee a strong authentication and X.509 identity and proxy certificates [80] in order to provide a globally unique identifier. GSI relies on trusted third parties for signing users and host certificates. It commonly uses the Transport Layer Security (TLS) protocol for authentication, and defines a common credential format based on X.509 identity certificates. Authentication is provided by the conjunction of an X.509 certificate and an associated private key. GSI certificates are issued by a trusted party called the Certification Authority (CA).
- *Credential management*. Certificate management is supported by MyProxy [55], a remote client–server system in which clients can store credentials with access control policies in an online repository for later retrieval. Typically, users store long-lived credentials in the repository, which are thereafter used for providing short-lived credentials for grid sessions. MyProxy uses X.509 certificates and can be combined with a Certification Authority service that automatically stores its own certificates, to be used for ID/password sign-on. A command-line interface is provided allowing users to introduce user ID and password to obtain grid proxy credentials on their local services.
- *Credential delegation and single sign-on*. These features are implemented with the help of proxy certificates [80]. Proxy certificates are similar to X.509 digital certificates except that they are signed by an end user, and whereas X.509 certificates are intended for assigning long-term identities, proxy certificates are used for short-term delegation of rights. They allow a user to create and delegate a set of credentials to another user without the involvement of an administrator. Users generate delegated proxy certificates with short life spans that get passed from one component to another and form the basis of authentication, access control and logging.

For single sign-on, the user signs in once in order to create the proxy certificate which is thereafter used for all subsequent authentications. Proxy certificates may be created locally by the user and used by subsequent clients. Thereafter the proxy certificate can be used to authenticate to a remote service. Proxy certificates can thus be used to create other proxy certificates, allowing thus a chain of delegations.

These delegations can also be restricted with the help of the policy specified in a restricted proxy certificate. Restricted proxies represent an extension of the X.509 certificates to carry restriction policies, enabling grantors to delegate only a portion of the rights they possess.

The format of the restricted proxy credential is neutral and as a result can support different policy languages.

- *Authorization and access control*. Authorization and access control was initially based on a simple access control list placed in a flat file called the Gridmap. GT2 used the Gridmap file to map a user to a user ID on a remote resource during job submission. GT3 extended this idea by allowing Gridmap files to be associated with services and factories, and consequently applying access control policies to those services and factories.

In a Web services context, the question is who is authorized to use a certain Web service. In GT4, GSI supports authorization in both the server-side and the client-side, and has six possible authorization schemes:

- none: no authorization is required
- self: the client may use a service, or authorize an invocation by the service, if client and service have the same identity
- Gridmap: a list of authorized users akin to an access control list
- identity authorization: a client will be allowed to access a service only if the client's identity matches a specified unique identity, or will only allow requests to be sent to services with a specified identity.
- host authorization: the client is allowed to access a service if it presents a host credential that matches a specified unique hostname; likewise, the client will authorize an invocation if the service has a host credential.
- SAML Callout authorization: the authorization decision can be delegated to an OGSA Authorization-compliant authorization service.

GSI supports also custom authorization by providing an infrastructure to easily plug into other authorization mechanisms.

2.2.1.1. Dynamic creation and management of overlaid trust domains. In order to establish a VO from overlaid trust domains, GSI uses both proxy certificates and security services such as the Community Authorization Service (CAS). GSI has as a policy that two entities bearing proxy certificates issued by the same entity will trust each other. Users may therefore create simple trust domains dynamically by issuing certificates to services that are intended to collaborate with each other.

2.2.1.2. Transport-level and message-level security. GT3 and GT4 provide mechanisms for both transport layer security and message-level security. The transport layer security is based on a GSI-enabled HTTP protocol, and message-level security on technologies and standards such as WS-Security, XML Encryption, and XML Signature. The TLS-based protocol provides also message protection (encryption and integrity checking). The following security schemes are provided:

- GSI Secure Message: Provides message-level security based on WS-Security, and supports privacy and integrity. The performance is good if only few messages are sent.
- GSI Secure Conversation: Provides message-level security based on the WS-Secure Conversation specification and is the only scheme that supports credential delegation. It does support privacy, integrity and anonymous authentication. Its performance is good if many messages are sent.
- GSI Transport: Provides transport-level security by using TLS and it is used by default in GT4. Performance is very good and privacy, integrity and anonymous authentication are supported.

3. Security services and mechanisms

3.1. Some definitions

The services that GREDIA will offer to its users should be secure. This can be guaranteed if those services call to other services which are security services. Services are implanted thorough mechanisms.

The definitions of service and mechanism are not standard and they always depend on the context where they are going to be applied. Thus, as our context is OGSA we will adopt the definition of service provided by them. We will give the definition of service and some other basic definitions.

- *Service*. A service is a software component participating in a service-oriented architecture that provides functionality or participates in realizing one or more capabilities [73].
- *Security service*. A processing or communication service that is provided by a system to give a specific kind of protection to resources, where said resources may reside with said system or reside with other systems, for example, an authentication service or a PKI-based document attribution and authentication service. A security service is a superset of AAA services. Security services typically implement portions of security policies and are implemented via security mechanisms [43].

Since the concept of service has been defined as a *component,* we will here define the concept of component:

- *Component*. An interchangeable part of a system that encapsulates its contents and defines its behaviour in terms of its public interfaces [58].

Another important concept is the following:

- *Security mechanism*. A process (or a device incorporating such a process) that can be used in a system to implement a security service that is provided by or within the system.

Some security services are audit, availability, confidentiality, non-repudiation, integrity or accountability. However, due to their importance for GREDIA we will concentrate on authorization, authentication and access control (see Section 3.3).

3.2. Authorization, authentication and access control in grids

Regarding to authentication, a central role is played by the Public Key Infrastructures (PKI), which enables secure and efficient acquisition of public keys. The most widely used PKI is the Public Key Infrastructure X.509 (PKIX) [59], defined by the IETF's PKIX Working Group. The challenge posed by the existence of multiple identities, within or across domains, and multiple authoritative sources of identity, is met by the standards for identity federation established by OASIS and the Liberty Alliance Project [10], which defines mechanisms for sharing identity information between domains. It has been adopted by the Internet2 project Shibboleth [70].

3.2.1. Access control

There are currently several access control models for collaborative environments. We briefly introduce them below.

- *Access Matrix Model*. One of the earliest access control mechanisms [67] mainly used for resource protection in an operating system, and which specifies in a matrix the rights that a set of subjects possesses for each object in an environment.
- *Role-Based Access Control (RBAC)*. In the RBAC model [68] permissions are assigned to roles rather than to individual users. RBAC is more scaleable than user-based access control and reduces the cost of security administration. RBAC models have been successfully implemented in workflow systems and distributed environments.

- *Task-Based Access Control* (TBAC). TBAC [74] extends the subject/object-based access control models by including domains containing task-based contextual information and allowing dynamic management of permissions as the tasks progress.
- *Team-Based Access Control* (TMAC). TMAC [75] extends RBAC by grouping users in teams, and introduces the notion of user context identifying users playing a role on a team at a given moment, and object context identifying specific objects required for collaboration purposes.
- *Context-based TMAC*. This model [33] integrates RBAC and TBAC by incorporating context as an entity in the architecture.
- *Spatial Access Control. SPACE* [11] is a *spatial access control* for collaborative virtual environments. It consists of an access graph and a boundary dividing the collaborative environment into regions. Credentials are used to allow access within regions.
- *Context-Aware Access Control*. This model [16] is an extension of RBAC with the notion of environment roles capturing environment state in order to provide for security in context-aware applications.
- *Attribute-Based Access Control (ABAC)*. This model [63], in which access decisions are based on the attributes of the requestor and resource, is becoming increasingly important, and has been adopted by several authorization systems such as Akenti [76], PERMIS [12] and VOMS [1]. Roles are a kind of attribute, and both RBAC and ABAC may be supported by the X.509 Privilege Management Infrastructure (PMI) standard [60], which extends PKI with attribute certificates to support tasks related to authorization.

3.2.2. Authorization systems and models

In this section we introduce the basic concepts and models of authorization as well as some of the authorization systems and mechanisms that are candidates to become OGSA standards.

Identity-based authorization systems have so far been the most widely used ones (e.g. Gridmap files [72]), but lately both role-based authorization and attribute-based authorization are emerging in grid computing. The Global Grid Forum [34] created an OGSA Authorization Working Group whose task is "to define the specifications needed to allow for inter-operability and pluggability of authorization components from multiple authorization domains in the OGSA framework" [35]. The objective of these specifications is to allow these systems to be interchangeably used with middleware that requires authorization functionality.

A number of authorization systems that are emerging in the grid today have been highlighted by the OGSA Authorization Working Group. These systems are candidates to become OGSA standards [71]. They include, among others, Akenti [76], CAS [61], PERMIS [12], VOMS [1], and PRIMA [51]. These authorization systems typically use assertions that bind attributes to users for purposes of authorization decision. However, there is currently no standard within attribute-based authorization for how attributes are communicated and for expressing policies regarding those attributes. Specifications are under development today that are intended to provide basic inter-operability among authorization components in the OGSA framework.

Typically, authorization decisions are based on information provided by authorities that are related either to the authorization subject and/or to the resource that is the target of the authorization request. Three basic entities are involved in authorization decisions: (i) *subject*, a person or process which is a user of a service resource; (ii) *resource*, an entity that provides or hosts services according to a set of access control policies; and (iii) *authority*, a trusted entity that may issue, validate and revoke several types of assertions, e.g. attribute, policy and identity assertions.

3.2.2.1. Authorization models. The following authorization models can be recognized [81,52].

- *Push model*. In a push model (Fig. 1) the subject requests an authorization from an authority (step 1), which acts as a service that issues authorization decisions in the form of authorization assertions that are returned to the subject (step 2), which may convey them to the target resource (step 3); the resource may then grant or deny the access request (step 4)

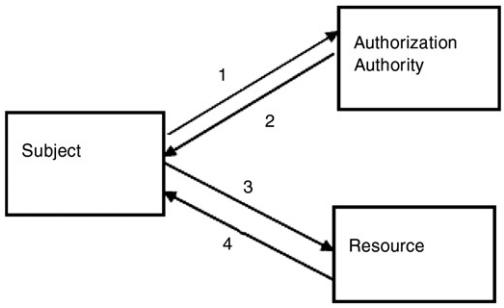


Fig. 1. Authorization push sequence.

- *Pull model*. In a pull model (Fig. 2) the subject sends the request directly to the resource (step 1), which subsequently contacts the authorization authority (step 2); the latter will then send an authorization decision to the resource (step 3), which decides whether the corresponding access request will be granted or not (step 4).
- *Authorization agent model*. In an authorization agent (Fig. 3) model the subject sends the access request to an agent (step 1) that thereafter makes authorization decisions according to the rules established by an authorization authority, whereupon the resource is contacted (step 2) for feedback (step 3) and subsequently a final decision is sent to the subject (step 4).
- *Hybrid model*. In a hybrid model we have a combination of some of the above models; an example is when the pull model and the push model are combined: the subject may obtain authorization to access a resource from an authorization authority, as in the push model, but the resource may also query an authorization authority in its local domain for compliance with local policies, as in the pull model.

In all the models above, the authorization service typically gives the descriptions of a subject, the action requested, and a target resource, and returns an authorization decision. The proposed message format for requesting and expressing authorization assertions from an OGSA authorization service is SAML. The goal with using standard message formats such as SAML is to allow different authorization systems to be used interchangeably in OGSA. Authorization in Akenti and PERMIS is based on attribute assertions from external sources, in VOMS on assertions of group membership, and in CAS on capability assertions from a VO server. In CAS, Kerberos and Keynote the credentials used are authorization assertions.

We can also divide grid authorization systems into *VO level systems* and *resource level systems*. The former has a centralized authorization system that provides credentials for users to access the resources. The latter, on the other hand, allows users to access the resources based on the credentials that were provided by the users themselves. VO level systems are CAS and VOMS, whereas Gridmap files, Akenti and PERMIS are examples of resource level systems (see below for a discussion of these authorization systems). Authorization systems at distinct levels may thus complement each other in an authorization system.

The scope of authority is called an administrative domain. A subject, resource or authority, belongs to one or several administrative domains. In a grid, the subject that requests a resource and the resource are typically in separate administrative domains, and there

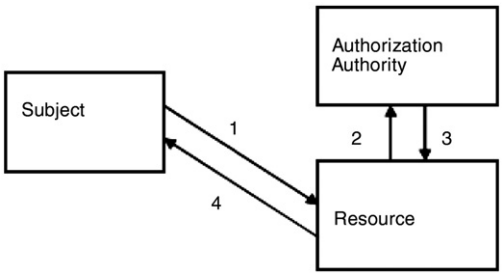


Fig. 2. Authorization pull sequence.

is a virtual organization domain that provides privilege management authorities for the VO members. Contractual relationships between different domains are often necessary which can be used for establishing trust relationships.

Authorization decisions are typically based on authorization information such as attributes, identities, policies and context parameters. There may be different authorities and domains for each type of authorization information.

Attributes and policies may be communicated during interactions or pulled from a repository, either local or associated with the attribute authority, and must be reliably bound to the issuing authority or the repository. Attribute certificates must be protected for integrity, authenticity and issuer non-repudiation, which is typically done by the making use of digital signatures, secure storage or secure communication. Signed SAML and X.509 Attribute Certificates are two common attribute certificate standards.

Authorization requests and responses must also be protected to ensure secure binding of a subject and the corresponding resource request as well as of the authorization response and the request. Secure channels, secure containers and signature mechanisms can be also used here.

In [81] two access control points are defined where access control functions are performed: *Policy Decision Point* (PDP) and *Policy Enforcement Point* (PEP). A PDP makes authorization decisions about a subject's access request to a service, and a PEP mediates access to a resource.

Authorization decisions are carried out according to a policy, usually stored in a repository or provided by policy authorities, which may be under the control of the resource owner or a topmost authority responsible for a separate Privilege Management Policy. Policies are usually expressed in a policy language and stored either in a single repository under the control of the PDP or another policy issuer, or distributed in a common VO domain.

3.2.2.1.1. The GREDIA platform. Data objects are central resources in GREDIA applications. However, in GREDIA the owner of a data resource, the publisher, does not store the resource locally but in the P2P overlay. Thus it cannot directly enforce resource access. It would be convenient here to distinguish between two kinds of objects within an ordinary data object: the *raw data object*, which can be an encrypted text; and the *content* of the data object, which might be hidden inside the data object by encryption. This adds some complexity to the authorization models above. Both the data server of the P2P overlay, and the publisher of a data object, may act as enforcements points: the P2P overlay by enforcing access to the raw data object, the publisher by enforcing access to the contents of the data object by means of encryption. The publisher may delegate access control entirely to the P2P data server, or store data in an encrypted form. In the latter case, the P2P overlay may be seen as completely transparent to authorization and all stored data as public. Any intermediate model is possible here. The publisher and the P2P data server may apply their own access control policies, or collaborate together to establish the corresponding access policies.

3.2.2.2. Authorization systems. *Gridmap files* uses existing local mechanisms for authorization by authenticating a user which is then mapped to a local identity by a local configuration file called the Gridmap file. The mapping is used as an access control check, allowing or denying access to the requested resource if the user is listed or not listed in the local mapping. After the mapping, authorization relies solely on local policy management and enforcement mechanisms, allowing the operating system to act as a sandbox.

Gridmap files is not suitable for GREDIA applications. The system is not scalable and lacks the required expressiveness. In GREDIA, the P2P would need to be responsible for the Gridmap files, which is scarcely feasible. For more complex trust domains there is a security service called the *Community Authorization Service* (CAS) [61] that supports the specification of more expressive resource access policies. CAS is presented in the next section.

The *Community Authorization System* (CAS) was developed in order to solve the shortcomings of the Gridmap files. CAS is designed to work together with GSI, and uses X.509 proxy certificates for providing authentication, single sign-on, delegation and credentials. CAS works as a push model and allows for separation of concerns between site local policies and VO policies by enabling sites to delegate management of a portion of their policies to the VO. CAS allows a VO to explicitly maintain and communicate its own set of policies, which may be combined with local policies. In order to obtain more consistent policies across domains, CAS provides a fine-grained mechanism for the management of the delegated policies, and supports the enforcement of these policies.

CAS supports also the notion of groups for users and resources, which allows the specification of roles and grants associated with these roles. The implementation includes a CAS server that is initiated for a community and acts as a trusted intermediary between VO users and resources. Resource providers typically establish trust relationships with the CAS server and then grant privileges to the community using local mechanisms, e.g. Gridmap files.

CAS is used to manage the community's trust relationships and grant fine-grained access control to resources according to the VO's access control policies. It holds information about CAs, a list of the users, server and resources within the VO, as well as policy statements specifying which users or groups have what permissions to access specific resources or resource groups. Permissions are expressed in terms of an action describing the type of action allowed, and a *service type* defining the namespace in which the action is allowed. The CAS server enforces also its own set of access control policies, e.g. concerning the rights to maintain groups with the VO or to delegate rights.

CAS assumes the existence of a virtual organization and in this sense it would thus be suitable for GREDIA applications. There are case studies that show the benefits of CAS applications involving a distributed network of storage systems, e.g. the Earth System Grid (ESG) [25] that contains environmental data. On the other hand,

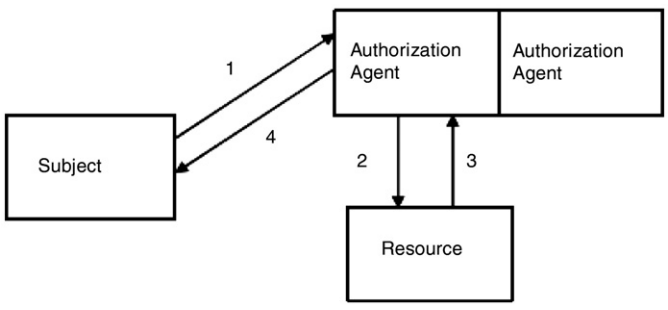


Fig. 3. Authorization agent sequence.

CAS was intended primarily for the scientific community, not for enterprise systems as envisaged by GREDIA.

In GREDIA, both the publisher and P2P data servers could enforce authorization according to presented CAS credentials. The capabilities denoted by the credentials should normally have been provided to the CAS by the publisher. CAS shows good scalability properties, essential for GREDIA applications. The main drawback is that a CAS server constitutes a single point of failure, unless there is some type of replication, i.e. by letting CAS servers be part of the P2P overlay. Also, CAS does not provide revocation mechanisms, which could be important in some GREDIA applications.

Concerning inter-operability, CAS is closely tied to the Globus Toolkit and uses GSI credentials, whereas many organizations use non-GSI credentials and standards like Web services standards or BPEL. However, some efforts have been made to integrate CAS with SAML and to use XACML for expressing policies. Thus, although GREDIA is oriented towards Web services and CAS may be considered as a pre-web service technology in the Globus toolkit, CAS can still be seen as a useful component for the GREDIA platform.

The *Virtual Organization Membership Service* (VOMS) [1] was developed by the EU project *14rid* [20] and the EU- USA project *DataTAG* [21] as a way of delegating authorization decisions to managers in the VO, since researchers were not satisfied with the functionality of CAS and the manually generated Gridmap file. The earlier versions had a mechanism for dynamically creating Gridmap files from LDAP directories containing information about users, freeing thus the resource owner from the task of creating the Gridmap file. Due to problems of scalability, a push system with no need for Gridmap files was devised, in which "pseudo-certificates" that the user should present to the resource owner were signed by the VOMS server. In its latest versions, the VOMS server produces short-lived X.509 user attribute certificates.

VOMS and CAS are very similar, and most features concerning applicability and inter-operability are the same. They both cater mainly to the scientific community. VOMS might be useful in GREDIA for applications requiring extensive use of roles and groups in a multi-VO setting. As in CAS, a VO is defined in VOMS as a community of users, institutions, and resources in the same administrative domain. Also, VOMS has a policy database for storing authorization policies. In VOMS, a user may be a member of any number of VOs. Each VO may consist of groups and subgroups for categorizing users according to their tasks. A user may be endowed with any number of roles and capabilities, according to VO or group membership. Certificates intended to be submitted to any VOMS system are obtained from a Certificate Authority. Credentials obtained from any number of VOMS systems may be thereafter provided to the resource.

PERMIS [12] is an open-source attribute-based authorization infrastructure written in Java and now part of the US National Science Foundation's Middleware Initiative (NMI). The key idea was to develop a system in which the resource administrator would be able to define the access policy and let this policy be enforced by the authorization infrastructure [13]. PERMIS is an attribute-based access control (ABAC) infrastructure, integrated with the Globus Toolkit to provide authorization for grid applications using the industry standard SOAP and SAML protocols. PERMIS defines a distributed Privilege Management Infrastructure (PMI). Attributes are stored in X.509 attribute certificates (ACs), as well as authorization policies written in XML. Roles are based on attributes. ACs can be stored in one or more LDAP directories. In PERMIS, managers throughout a VO can act as attribute authorities with a private signing key and a corresponding X.509 public key certificate. Resource owners may state which attribute authorities to trust when setting their own access control policies and then leave the authorization infrastructure to enforce it. A *Privilege Allocator* is used to construct and sign ACs, and to store them in LDAP directories.

PERMIS can operate in both push and pull modes. The subject wishing to access a resource starts by contacting the PEP, which must authenticate it. In the push mode the user pushes X.509 ACs to the application gateway, whereas in the pull mode the PERMIS PEP pulls the user's X.509 ACs from LDAP directories. PERMIS does not make any assumptions about the authentication method employed. PERMIS assumes a RBAC model where each user is mapped to a role and the policies are assigned to roles. After authenticating the subject in an application dependent way, and evaluating the subject's ACs according to the policy, rejecting untrustworthy ACs, the PERMIS PEP will contact the PERMIS PDP and forward the valid ACs for an authorization decision. This may be done either by a Java API invocation, if the PDP is built as part of the application gateway, or as a SAML request over SOAP and HTTP, if the PDP is built as a stand alone server. The PERMIS PDP makes the authorization decision based on the user's attributes obtained from the user's ACs, the authorization policy in force, the roles currently valid for the user, the policy for the resource, the requested action, and context variables such as the time of the day. The PERMIS PEP will then enforce the decision, which is a simple Boolean, on behalf of the resource.

In the Java API, the Constructor method is used to build the PERMIS decision engine, and take as arguments the X.509 Source of Authority, which is the root of trust for authorization, a policy identifier, and a list of LDAP locations for retrieving the ACs. The *GetCreds* method fetches the ACs from the roles of a user, and it is passed to the PERMIS PDP. The *Decision* method is used to grant or deny a given action and returns a Boolean.

The Privilege Allocator tool is normally used for attribute acquisition. It creates X.509 ACs and stores them in an LDAP server. Other software tools that may create ACs are a user friendly graphical Attribute Certificate Manager, and a bulk loader tool designed to allow large numbers of users to be automatically allocated ACs.

PERMIS allows two kinds of attribute models, a push model and a pull model. In the attribute pull model, the PERMIS PDP retrieves all the required X.509 ACs from LDAP servers, whereas in the attribute push model it is the PERMIS PEP that obtains the required AC's in an application dependent way, e.g. from the subject, and then forwards them to the PEP.

The PERMIS policy, written in XML, supports hierarchical RBAC, in which roles and attributes are given access rights. Superior roles or attributes in hierarchy inherit the privileges of inferior ones. The policy is created by the Source of Authority (SOA) for the resource, and stored as a digitally signed CA in an LDAP directory. During construction time the PERMIS PDP retrieves the policy and makes all decisions internally, hence no external agent can see the policy. Only a grant/deny response is then returned to the PERMIS PEP.

PERMIS is integrated with the Globus Toolkit through the use of SAML, and similarly to Akenti. PERMIS would be very adequate for many types of possible GREDIA applications, especially those relying on the RBAC model. It should be noted though that the access control API as well as mechanisms to allocate privileges is proprietary.

Other authorization systems such as *Cardea* [48], *Akenti* [76] and PRIMA might also be relevant for grid applications, but will not be presented here.

3.3. Trust management

From the perspective of grids a fundamental idea in the definition of grid is *resource sharing* [27]. The first grids were developed for partners that need to collaborate in a specific task and they knew each other well. In this case there was an implicit trust relationship among the partners. However, the grids have been lately used mainly for business purposes (this is the case of the GREDIA scenarios) where the partners may not know each other and even though they have to share resources. In these cases trust mechanisms are needed.

Trust management systems can be centralized or decentralized. In centralized systems there is a central authority that manages the credentials and distributes them among all the components of the

system. Distributed approaches are more appropriate for grids as grids are distributed systems.

There are several classifications of trust management systems [36,44]. For instance, *Service Provision Trust* describes the trust of a relying party in a service or resource provider. This is one of the types of trust considered in these classifications and it turns out to be essential in grids. But the most interesting classification of all for our purposes within GREDIA distinguishes between credential-based trust management systems and behaviour-based trust management systems. Depending on the way of establishing or evaluating trust one of these methods becomes more convenient.

Concerning grids, credential-based trust management systems are the most widely used. This type of trust is present in the inclusion of certification authorities for the authentication mechanisms that guarantee that the nodes of the grid belong to the trusted organization that participates in the computation. But as the scope of grids is growing and they are evolving to ubiquitous or pervasive computing, there is a need to keep the reputation of the node once it has been allowed to enter the grid. Thus, behaviour-based trust management systems become also crucial for grids.

3.3.1. Trust management for grids

Providing trust for grid architectures has been a hard problem to tackle and there are no many trust or reputation systems for these systems. Some efforts have been directed to provide trust using security mechanisms. Trust and security within grid environments are focused on the need to provide scalable and dynamic VO. The work presented in [50] considers a way to enhance security in GSI by using two level trust models. On one hand, they define a distributed trust model for the VO and, in the other hand a centralized model for each domain in the grid.

Since grid architectures are evolving to P2P systems and business usage, some works have used reputation systems for grids. Some of these systems are GridEigenTrust [47] and PathTrust [45]. GridEigenTrust is an extension of EigenTrust for using it in grids. Grids include virtual organizations and this provides an obvious classification of resources, users, and their reputation that is needed in order to establish scalability. They introduced an implicit hierarchy of entities that belong to organizations, and organizations that form virtual organizations. Thus, the authors assign reputation values first in the lowest level and go up in the hierarchy until they finally compute reputation of a virtual organization. The trust of a virtual organization will be computed based on the trust of its organizations and the trust of the organization will be computed using the trust of its entities.

PathTrust is a reputation system developed for member selection in the formation phase of the virtual organization. The PathTrust algorithm arranges the participants in a graph such that each edge in the graph is weighted with trust between the nodes at the end of the edge. Trust is computed by accounting the number of positive feedbacks and subtracting the number of negative feedbacks weighted by the total of positive and negative feedbacks obtained. Other approaches deal with trust in virtual organizations. In particular, the authors present a trust-based access control model for virtual organizations. This model is based on the Shibboleth technology. Trust relationships are established between three different parties in the VO (users, groups and vroom; a Shibboleth Identity Provider and a Shibboleth Service Provider). Depending on the trust relationships between parties the algorithm determines the access permissions.

4. GREDIA security framework

4.1. GREDIA architecture

As we have mentioned above GREDIA aims to enable commercial users to manipulate data and access services in a grid computing environment. Several components have been identified within the GREDIA architecture [18] (see Fig. 4). These components are the following:

- *The Framework for Intelligent Virtual Organizations (FiVO)*: This system enables participating users to join virtual organizations by agreeing to and signing *virtual organization contracts*. These contracts specify the duties and permissions of each VO participant as well as the resources which can be used by that group of participants. A suitable interface will be developed for creating and modifying the contents of VOs.
- *The Application Development Platform (Appea)*: The core system of GREDIA, this is a platform which enables specialized users (whom we call Application Scenario Developers) to prepare *application scenarios* that can be executed in the GREDIA architecture. This is done by writing *application scripts*, which can make use of *grid services* and *data sources* (present in the infrastructure) as well as call upon *actors* who are registered with the VO framework to perform certain tasks related to the business processes which they model. Appea provides an integrated, powerful environment for manipulation of Grid Objects (i.e. services), data sources and actors while freeing the scenario developers from details related to interfacing and invoking the actual underlying software modules.
- *The Grid Run Time Service Discovery Tool (GRSD)*: This component is used to support the identification of services that can fulfil functional, non-functional and contextual characteristics of the application scenarios. More specifically, the runtime service discovery tool identifies services that can replace services participating in the application scenarios that become unavailable, fail functional or quality service requirements, or should not be used due to changes in the context of the services or the context of the application scenarios. The services are identified based on synchronous and asynchronous queries specified by the Appea Runtime System.
- *The Rich Data Location Service (RDLS)*: A peer-to-peer framework for storing and accessing annotated multimedia content in a distributed environment. This component will enable users of the system to upload and download multimedia files using P2P algorithms, as well as query for the types of files which might interest them in their work. Suitable metadata descriptions will accompany each file, thus facilitating query processing.
- *The user portal and clients*: In order to access the functionality of GREDIA, as provided by Appea, RDLS and other client-oriented modules of the system, a user portal is being developed, capable of interfacing with lower layers of the GREDIA infrastructure, handling interactivity and executing application scenarios. For this portal, two types of clients are envisioned — a client for stationary PCs and a mobile client, tailored for mobile devices.
- *Mobile Proxy*: Services which are deployed on and exposed from mobile services will be accessible through the use of a special stationary Mobile Proxy. This proxy will keep track of mobile clients and will be capable of contacting them in an asynchronous way. In essence, the Mobile Proxy will represent mobile services to other components of the GREDIA infrastructure.

Security should be provided for each of these components. The security framework is a horizontal component of the whole GREDIA platform and should be embedded in each of them in such a way that it is done as part of their deployment. We will show how this is done in the following sections.

4.2. GREDIA workflows security

The GREDIA platform is intended for workflow-based grid applications. Workflows may be described as executable business processes. Being the focal point in the GREDIA architecture, it is important that its security aspects are considered in detail.

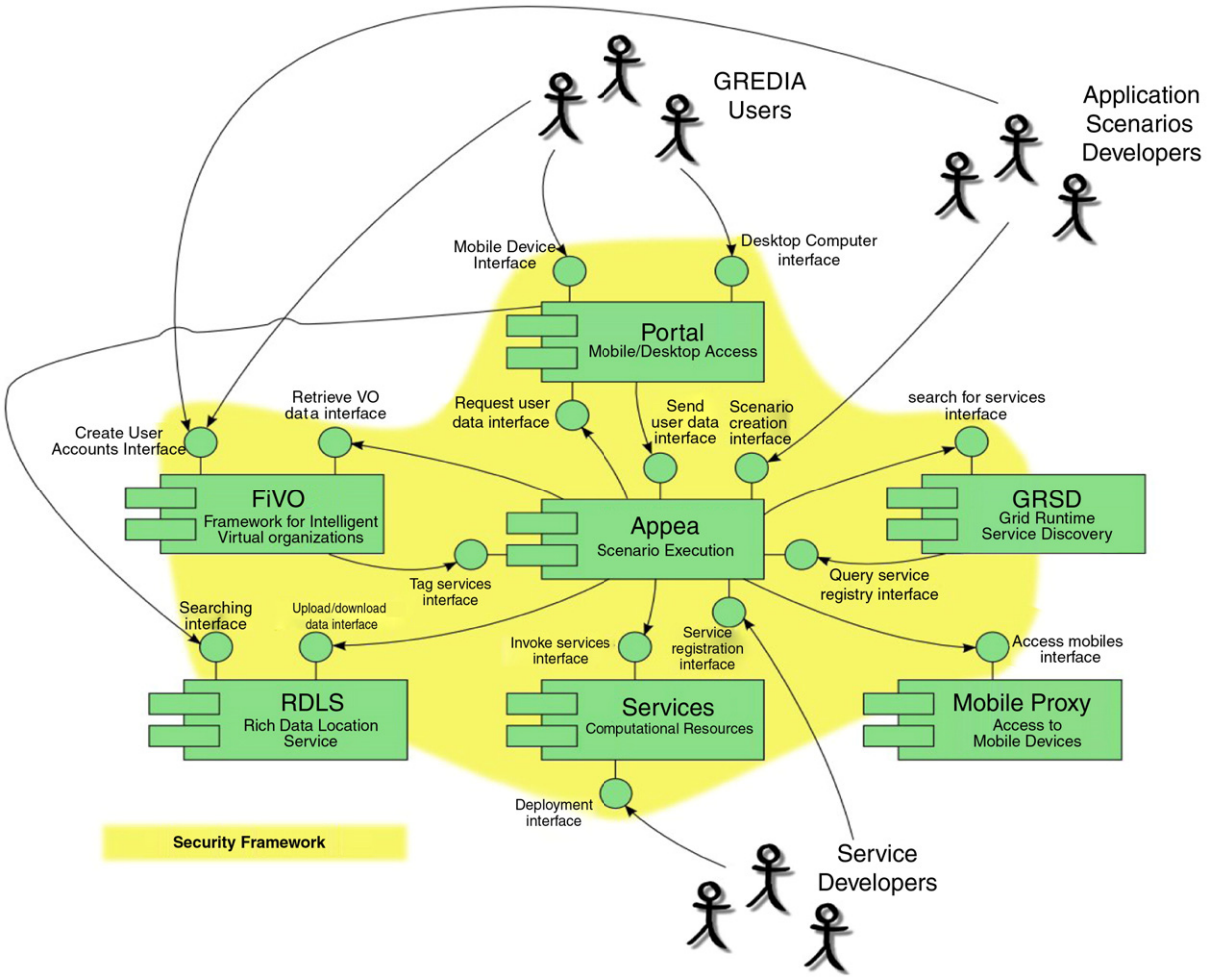


Fig. 4. GREDIA architecture components.

The concept of workflow or business process is of paramount importance in modern information technology. Business processes have been usually considered as the starting point for system development, providing a simplified view of business structure and the system requirements necessary to support the business. Today we may consider them also as the result of the system development process, as a type of executable code. As a result, business process security has become a crucial aspect of modern information technology, and many systems will only be deployed only if enough assurances are given regarding their security properties.

Moreover, business process can be profitably used in the requirements phase of a system development, and might serve as the link between this phase — which is not part of the GREDIA framework — end the implementation phase of a GREDIA application. Business process models offer also a good opportunity to the early introduction of security aspects in system development, which is one of the objectives of the GREDIA project. For instance, UML activity diagrams enriched with security information could be mapped to GREDIA workflows. As it is widely recognized, there is little systematic support for software engineers to design secure systems, and the specification of security policies is typically not integrated with general software development. Security is usually added as an afterthought to system requirements and design.

Modelling business surroundings involves answering the following questions:

(i) how do different actors interact;
(ii) what activities are part of their work;
(iii) what are the ultimate goals of their work;
(iv) what other people, systems or resources are involved that do not show up as actors to this specific system; and
(v) what rules govern their activities and structures.

The answers to these questions are important for the security aspects of a system. These questions can be included in business process representations but, except point (ii), not in workflow notations. We propose thus that they be represented in a suitable ontology language attached to GREDIA workflows.

A workflow or business process may be defined as a set of coordinated activities or *tasks* that achieves a business objective [4]. A task is a logic step or description of a piece of work that contributes towards the accomplishment of a process, and may be carried out by any processing entity, either human or automated. Tasks are related to each other through task dependencies expressing concurrency, serialization, exclusion, alternation, etc. A workflow management system (WFMS) supports the execution, monitoring, coordination and administration of workflow processes. Workflows may also involve many elements in the business surroundings which are relevant for security. These include agents performing the tasks, either people or processes, roles representing rights and artifacts such as procedures, information, information flows and material. Security requirements may concern any of these entities.

Role-Based Access Control is very common in workflow authorization models, and will be adopted in GREDIA. Organizational

workflows can be very large, with several thousands users. The complexity of access control is reduced by using RBAC since the number of roles is significantly smaller than the number of users. Typically, in RBAC roles are assigned to users based on their qualifications, and tasks in turn are assigned to roles.

We concentrate here only on the security aspects and requirements which are specifically related to workflow, e.g. separation of duties, object access authorization, inter-workflow security and security sub-processes.

Object access control implies that authorized users should gain access on required objects according to granted privileges and usually only during the execution of a specified task. Authorization mechanisms must ensure that the various tasks of a workflow are executed by authorized users. Hence, granting and revoking privileges should be synchronized with the progression of the workflow from task to task. We intend to introduce a suitable formalism capable of specifying authorizations allowing access control only during the execution of a task.

Separation of duties means that a person responsible for a determined task in a workflow may not be responsible for another specific task in the workflow. If there is no support for constraints within a workflow management system, these must be implemented as application code or within the tasks themselves, which is very awkward. We aim at introducing a formalism for the specification of separation of duties to be used by the workflow management system.

Inter-workflow security focuses on the security of the communications of workflows which might be located in different administration domains. Important security issues here are audit, authentication, secure key exchanges and the management of distributed security profiles, all of which must be provided by the security framework.

Security sub-processes [41] is a much neglected aspect of workflows. Workflows usually concentrate on modelling the functionality of a system, disregarding security considerations, a fact which often leads to serious security vulnerabilities since security is thus left to be integrated into an application at a later stage of software development and in an ad-hoc manner. Many proposals have thus been presented to express security semantics within business processes and workflows. Security sub-processes specify a sequence of steps that is necessary to insert into a process for security reasons, for instance for authentication or verification of authenticity of signatures during contract signing. This sequence of steps constitutes a workflow in itself, can be reused, and also viewed as a security pattern that can be inserted into a workflow whenever required by the security requirements associated with the workflow.

Although several methodologies have already been proposed to deal with workflow security, none of them are adequate for real world applications, and current commercial workflow management systems do not provide designers with support for the easy integration of security goals within workflows. The GREDIA project offers a unique opportunity to develop and test new techniques and methodologies in this field. The focus should be on the establishment of security patterns at the workflow level and of clear security ontology with the help of a high-level modelling language that is at least inter-operable with the ontology languages used in the GREDIA component FiVO and service descriptions.

The most critical components for adding security features to GREDIA workflow specification and management systems are:

- *Execution Service*: the Execution Service should be able, whenever required, to
  - authenticate application users
  - execute application scenarios according to the security constraints associated with them
  - maintain state information necessary to enforce related security constraints
  - present the required credentials to invoke operations on other services within the framework
  - present the required credentials to perform queries and retrieve data from the Rich Location Service
- *Developer GUI*: the environment of the application scenario developer should give support for the specification of security features in the design of application scenarios; the script language should be extended with the capability to express security requirements and constraints.
- *Scenario Repository*: Each scenario application should contain information regarding security requirements and constraints.
- *Invoker*: the Invoker should be able to take security constraints into consideration when mapping invocations of generic services onto invocation on concrete implementations of these services.
- *Data Access Client*: the Data Access Client should be able to present the required credentials to perform queries and retrieve data from the Rich Location Service and other standalone database and external repositories.
- *Service Registry*: The Service Registry should provide means to include security semantics in both technology-independent and technology-specific resource descriptions. The developer should be able to take security goals and requirements into consideration when browsing registry data and searching for information about concrete services. Registry data should include security features.
- *RSM Registry*: It should be possible to integrate security semantics within the overall service semantic information.

4.3. Virtual organization security

Fundamental to grid computing is the notion of scalable virtual organization (VO) [27], which may be defined as a dynamic set of individuals and/or institutions that share resources and services according to a set of well-defined rules and policies. The grid vision is to provide unlimited power and information access to end users through the creation of dynamic VOs for secure and agile resource sharing among individuals and organizations. VOs may span several administrative domains, each one with its own security requirements and policies. Hence, inter-operability among the multiple domains involved in a VO requires that VO-defined policies comply with domain-level policies, while at the same time maintaining a clear separation among virtual and real protection domains in a context in which they may superpose and intersect each other in a variety of ways.

The virtual organization of GREDIA is called FiVO (Framework for Intelligent Virtual Organizations). As we mention earlier.

Trust and security are especially important in the deployment of the VOs, more specifically authorization, delegation and authentication. Each virtual organization usually goes through several phases:

- Creation
  1. Identification of market opportunity
  2. Partners identification
  3. Contract negotiation
  4. VO deployment
- Operation/Evolution
- Dissolution. Provenance of obtained results.

Within the creation phase perhaps the most interesting stage is the contract negotiation. At this point the policies and agreements among partners are specified. This means that knowledge of the domain is essential. This knowledge is encoded in the form of ontologies.

Regarding to authorization, GREDIA will use PERMIS as the authorization service. Therefore, including the specifications of PERMIS within the ontologies used by the definition of the FiVO is of vital importance.

During the creation phase of the VO trust plays an important role in the partner's identification and contract negotiation phase as part of the decision-making process. These decisions could be based on the information acquired from previous interactions with the participants of the VO, i.e., on the reputation of these partners.

4.4. Security in service discovery and service description

The Web Service Discovery process should address both the security semantics of service descriptions as well as the general security and trust requirements associated with service discovery [78]. However, security problems are usually not addressed during the service discovery itself. Security is also commonly associated with the services but not with the clients.

4.4.1. Semantics of service descriptions

With respect to the security semantics of service descriptions, ontology-based descriptions may be conveniently used. However, current Web services discovery solutions like UDDI [84], WS-Discovery [83] and OWL-S [53] do not address most security and trust issues. [79] discusses how WS-Discovery can be extended to cover confidentiality and privacy in service discovery.

WS-Policy [6] provides a general purpose model and syntax to describe and communicate the policies of a Web service. The Web Services Security Policy Language (WS-SecurityPolicy) [56] defines a set of security policy assertions which apply to Web Services Security. Furthermore, WS-PolicyAttachment [7] defines several methods for associating the WS-Policy expressions with Web services. WS-SecurityPolicy is designed to work with the general Web services framework including WSDL service description, UDDI, and SOAP message structure.

4.4.2. Security and trust in service discovery

In a *centralized discovery approach* there is a registry playing the role of a yellow page which must be considered as a trusted third party by both clients and services. In a *decentralized discovery approach*, on the other hand, service discovery relies on peer-to-peer advertisements and direct exchanges between clients and services.

Possible threats in service discovery include denial of service and man-in-the-middle attacks. The registry may also become unavailable by flooding registration messages. Service lookup queries may also reveal the intentions of a client and thus have an impact on privacy requirements. Masquerading attacks in which a malicious agent fakes the identity of the registry, as well as replay attacks, may also expose important confidential information. Messages to a registry, both client requests and registration messages, may also be modified or dropped. Moreover, services may be registered, deregistered and discovered by unauthorized parties.

Possible countermeasures to these threats include the use of signed certificates to authenticate both servers and clients, blacklisting of untrusted parties, setting up secure channels, use of message authentication with signature or authentication codes, redundancy mechanisms to guarantee message delivery, adding signed sequence numbers to registration messages, registry authentication, signature of registration requests, use of restrictive policies obstructing service discovery by unauthorized parties, and so on.

The registry may act as a kind of early authorization decision point by restricting the clients that will be able to contact a service and hiding the service profile from unauthorized users. Discovery messages may include certificate, key or token credentials authenticating the client [79].

In decentralized solutions, where a trusted third party is absent, a similar level of protection may be enforced by using encryption schemes such as attribute-based encryption [62,66] or policy based encryption [5]. In these schemes, the user may encrypt messages according to a policy, and only users with the appropriate credentials will be able to decrypt them. In [77], attribute-based encryption is used to protect sensitive information included in the discovery messages.

In [17] a solution to secure service discovery was proposed. In this solution an entity provides a Service Discovery Service playing the role of a secure information repository or registry. Another early work targeting the security aspects of service discovery can be found in [87]. Both works focus only on the confidentiality of the messages during the discovery process. Service discovery security requirements and threats to service-oriented architectures using registry supported service discovery were presented in [77]. Finally, a work addressing privacy protection aspects of the discovery service was presented in [88].

4.5. Security in peer-to-peer service architecture distribution

Peer-to-peer architectures are characterized by their ability to function in the presence of a highly transient set of nodes, network, and computer failures without the need of a central server [3]. Instability and transient connectivity is thus the norm.

Ensuring content security in distributed peer-to-peer environments is challenging. The owners of data resources may send their data to data servers for further distribution according to the security policies associated with the data. The advantage of this scheme is that space requirements are removed from the data owners, making this approach specially appropriate to devices with limited storage capacity, as sensors, Smartphones, PDAs, etc. On the other hand, this feature also separates enforcement of access control policies from the storage of data and removes from the owner of the data resource the capacity to enforce access. Typically, authorization systems assume that the resource owner has the capability to enforce access control decisions, whereas access control decisions may be taken separately by both, the resource owner and the virtual organization according to specific policies. The latter still holds true for peer-to-peer content distributions systems, but the PEP is now controlled by the virtual organization. In this case, we may say that it is the VO that owns the resource. The security concerns that arise from this situation may be solved either by assuming that data servers are trusted, or by using cryptographic methods allowing the original resource owner to decide who will be able to read the data.

In an open environment, the first solution might be inappropriate. However, in virtual organizations, where the participants are bound by a contract, these solutions might be adequate by imposing a trust model where data servers are given a trust level corresponding to their functionality.

Harrison and Jensen [40] propose a cryptographic access control scheme that relies exclusively on cryptography and that provides both data integrity and confidentiality. Files are stored in an encrypted form and access control is enforced by distributing the corresponding symmetric keys.

4.5.1. Publish/subscribe systems

Security for published and stored content relies on cryptographic algorithms and protocols. Secure storage, secure routing, availability, privacy, confidentiality, integrity, authentication and access control are security goals that represent particular challenges in peer-to-peer content distribution architectures [3].

In a publish/subscribe system typically, a publisher publishes an item through a broker, which is responsible for dispatching this event to subscribers which have subscribed previously with the broker to this specific kind of event and have been authorized to receive it.

Availability and confidentiality requirements in the context of these systems typically imply that events are made available to authorized subscribers and only to authorized subscribers respectively. Thus, key management plays a crucial role here, as well as efficient cryptographic methods.

4.5.2. Secure storage

Self-certifying data can be used to ensure integrity. This is done by calculating a cryptographic hash to produce a file key, cf. the CFS system [19] and PAST [24], based on signed files and use a hash-derived signature as the file's access key.

Information dispersal algorithms, first considered by [64], are also used for secure and reliable content storage. A file is split into n pieces

in such a way that it can be reconstructed by any m pieces, where $m \leq n$. Systems that use this method are Mnemosyne [39], Publius [82], and FreeHaven [22].

Secret sharing, e.g. Shamir's Secret Sharing Scheme, is also used by several systems (Publius [82], PAST [24]. MojoNation [54]). A file is encrypted with a key which is split into n shares such that any m shares, such that $m \leq n$, but no $m-1$ shares, can reproduce the key.

Anonymity may be provided by Anonymous Cryptographic Relays, a mechanism used in PAST [24] in which a publisher sends via an anonymous connection encrypted shares of a file to several selected forwarders, which thereafter forward the shares to other selected nodes acting as storers for the shares. The contents may thereafter retrieved by clients by contacting the forwarders, which will then contact the corresponding storers which decrypt the shares and send them back to the client.

In Distributed Stenographic File Systems [2], employed in Mnemosyne [39], blocks together with random data are encrypted and then placed at pseudo-randomly chosen locations, making them undetectable.

Erasure Coding is a mechanism implemented by OceanStore [65] in which data is broken into blocks and spread over many servers, in a way that only a fraction is needed to reconstruct the original block.

4.5.3. Authentication and authorization

In peer-to-peer systems with highly transient nodes that employ content replication or fragmentations schemes, a particular security threat in the presence of several identities for the same physical entity is posed by the Sybil Attack [23]. The only effective countermeasure here is the presence of a central identification authority. Identification challenges may also be employed but are much more resource demanding and assumes that the resources of the attacker are limited.

In the absence of authentication, distribution of keys to a set of privileged users may be used for allowing content access. Access control can be enforced by the use of access control lists (ACLs) that are assigned to objects by their original authors through the use of signed certificates, as in OceanStore [46]. ACL entries may also extend privileges by describing the privilege granted to users. All content modification is verified against the ACL assigned to the object.

4.5.4. Anonymity

Anonymity requirements may refer to the content publisher, the identity of the storing node, content identity or metadata and retrieval query details.

An approach to the provision of anonymity is the dissociation of content source and requestor, as implemented in Freenet [15]. It employs a mechanism to make it unfeasible to track the true origin or destination of a file whereby a file is passed from the node that holds it to the originator of a request through each node that forwarded the request, and by allowing any node along this path to claim being the source of the data.

Another approach is by the use of infrastructure for providing anonymous connection layers, e.g. onion routing [37], mix records [14,8], or Freedom anonymity system [30]. Serjantov [69] uses a scheme in which documents are split into encrypted shares stored in nodes that are selected by an anonymizing layer of nodes called forwarders that construct “onions” around those nodes and anonymously forward the shares together with their anonymous return addresses. A similar system is FreeHaven [22]. The Tarzan system [29], on the other hand, is a decentralized anonymizing network layer infrastructure that allows a client to communicate with a server anonymously to other entities by building anonymous IP tunnels between an open-ended set of peers.

4.5.5. Encryption

Multimedia security focuses on protecting video, audio and image data. This is made possible by allowing safe communication or by protecting the multimedia against piracy, for example, using watermarks.

When the multimedia data is static (i.e. it does not need real-time streaming) it can be treated as binary data and can be encrypted using conventional algorithms such as DES or AES.

Nevertheless, the multimedia encryption has to face, on the one hand, the excessively large quantity of information [9] and on the other, the need for the multimedia data to be processed in real-time (the speed for MPEG-2 can be greater than 40 Mbps). When the video and audio data need a high transfer speed, for example real-time streaming, the previous solution is not suitable. Conventional cryptosystems are too slow to process multimedia data, and therefore the security of many multimedia applications in real-time cannot be ensured. As a result, selective encryption [31] is necessary where only some portions of the bitstream are encrypted. These portions will be selected according to the data compression format. How the multimedia encryption algorithms access the data depends on the compression format, for example, in the case of the MPEG video format, headers, frames, vectors of movement, etc. are encrypted using algorithms such as DES or AES.

To improve multimedia encryption algorithms, mechanisms could be applied to increase the speed of encryption although this would be detrimental to the security. Given that the multimedia data does not generally need high levels of confidentiality, this would be a good solution for journalistic information, since its monetary value is not so high as to make it attractive to attacks, and a lightweight encryption (degradation) would be sufficient. On the other hand, information related to industrial, governmental or military secrets needs a high level of security, so highly secure encryption algorithms would be applied to the detriment of the encryption speed. Therefore, these algorithms should be implemented according to the different security levels, both in the grid by means of GSI extensions, and in security libraries for wireless connections.

There is a wide spectrum of multimedia applications with different security requirements available, ranging from military applications that need a high level of security to applications needing to see the data in order to view and search a database. The characteristics that these algorithms must fulfill are [49]:

- They should provide sufficient security for every application.
- They should substantially reduce the computational cost with respect to total encryption.
- They should produce a bitstream compatible with the standard formats.

4.6. Security for mobile applications

Mobile devices have limited capabilities, which must be taken into consideration when their security mechanisms are designed. For devices with limited computational capacity, for instance, public key cryptography would not be a good option. It should also be pointed out that these are personal devices and, therefore, users taken them with themselves everywhere, considerably increasing the risk of theft or loss, with the resulting security risk.

Mobile devices exhibit some security problems due to their features:

- The physical weaknesses and limitations of wireless communications, e.g. high error rate, have an impact not only on features such as performance, but also security.
- The completely exposed environment associated with wireless air radio devices provides much more opportunities for malicious attacks and is more prone to accidental interferences.

The suggested security architecture for mobile devices is an attempt to improve some of their exhibited weaknesses. It is endowed with, among other features, a pseudorandom number generator (PRNG), support for secure storage, client authentication with certificates, and code signing. Furthermore, this architecture adds new

security services to mobile devices. Its API provides a set of security functions for mobile devices which will facilitate the integration of security aspects within mobile applications.

This architecture is build up of several layers (Fig. 5):

- Application security: This layer provides a high-level security API for wireless application programmers.
- Security services: Security services use protocols, policies and high-level security functionalities in order to provide protection to resources.
- Security management: The security management layer provides the maintenance of the security elements. The maintenance of an element involves tasks such as creation, deletion, modification, etc.
- Security utilities: This layer provides very specific functionalities, which require a special treatment.
- Security data: This layer provides storage capacities to security data.

The integration of mobile devices into the grid raises some problems [Isa05]. Although the connection does not require modifications in the existing infrastructure, it might undermine the operation of the grid due to the high level of error that these devices present.

In order to avoid it, mobile devices could access the grid via a proxy, which would be responsible for interacting with the grid on behalf of the device that it represents. In this way, the mobile device could delegate functions involving a high computational cost to the proxy. On the other hand, the proxy is not only a potential performance bottleneck, but it represents a "man-in-the-middle" which has implications for the privacy of the communication. Though the data are encrypted, any person with access to the proxy might see the data in cleartext, and this would be unacceptable for applications that need a high degree of security, such as banking and military applications. To avoid this, a trusted proxy could be established.

5. The integrated security architecture

In the following, we will present the integrated security framework. Fig. 6 shows this framework and also what extensions of GSI are needed.

In this figure we include three different areas: the mobile side, the proxy, and the whole grid.

The modules on the figure contain one or two different blocks. Lighter colours mean existing GSI functionalities, whereas dark blocks mean extended GSI functionalities.

The extended GSI architecture contains four layers:

- Security services: Security services use of protocols, policies and high-level security functionalities in order to provide protection to resources.
- Security management: The security management layer provides the security elements maintenance. The maintenance of an element involves tasks such as creation, deletion, modification, etc.
- Security utilities: This layer provides very specific functionalities, which require a special treatment.
- Security data: This layer provides storage capabilities to security data.

Next we will explain in more detail the elements included in these layers.

Confidentiality is provided by GSI at two levels: the transport level and the message level. The GSI Transport provides transport-level security by using TLS, and it is used by default in GT4. The message-level security is based on the WS-Security and WS-SecureConversation specifications. Both security levels lack multimedia encryption algorithms. Thus, multimedia encryption algorithms, and protocols to exchange the related encryption information such as keys, algorithm names, etc., must be implemented. Data encryption algorithms such as AES or DES, which will be used in multimedia encryption algorithms, will be provided by third-party libraries (e.g. BouncyCastle libraries).

The Authentication method of GSI is based on PKI. All users and hosts have their identity certificates in X.509 format. Nevertheless, the mobile client authentication is not implemented in SSL of Java MIDP2. An application-level mobile client authentication protocol using identity certificates will be implemented. Fig. 7 shows the suggested protocol.

The mobile client sends its X.509 identity certificate to the proxy server. But the certificate does not prove the mobile client identity (certificates do not need to be kept secret, thus anybody could have

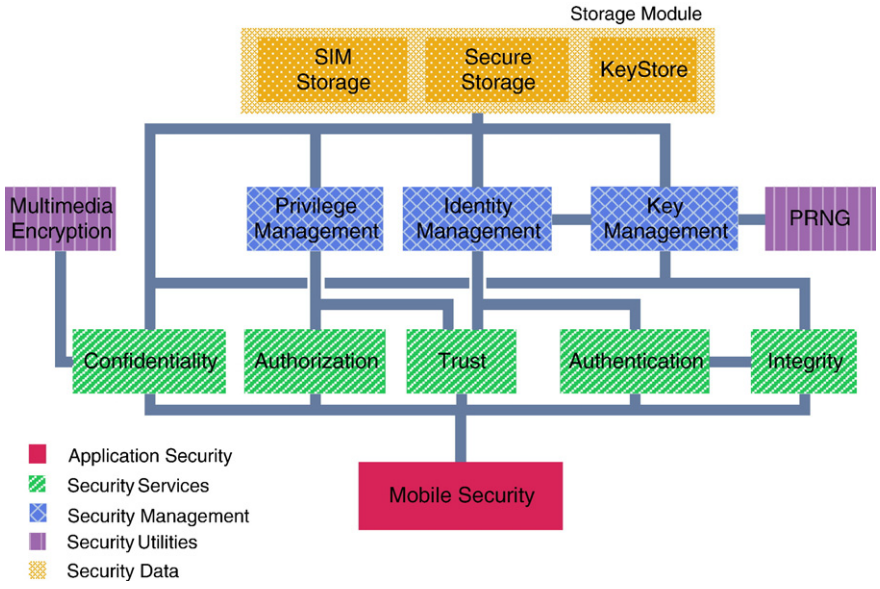


Fig. 5. Security architecture for mobile devices.

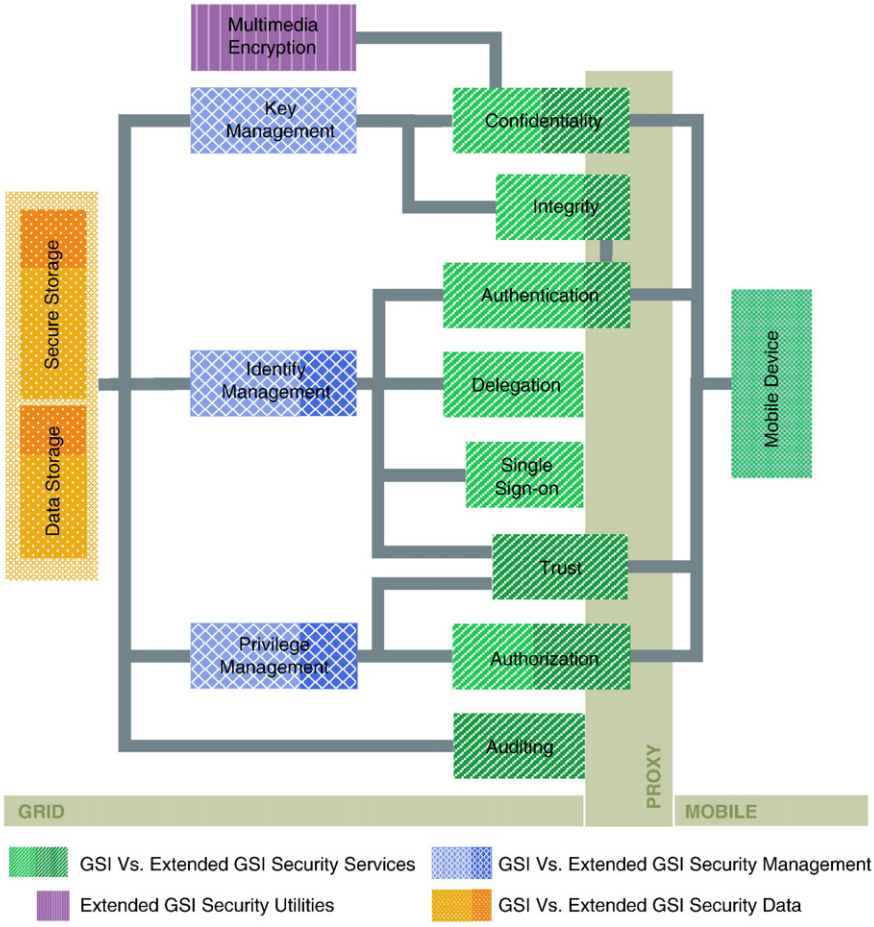


Fig. 6. Grid security framework.

the mobile client certificate). Thus, signed data is also sent to the proxy server. The mobile client uses his private key — which is only known by itself — to sign data, and when the proxy server receives the authentication response it verifies the signature with the public key included in the X.509 identity certificate. In this way, the mobile client proves its identity to the proxy. Furthermore, the proxy server verifies the identity supplied by the mobile client by using identity certificate X.509 searching in an access control list.

GSI provides functions to calculate and verify the message integrity-signed messages. These functions work within the GSI environment because they use GSS security context arguments (GSI implements GSS–API interface). But these functions are not suitable outside the

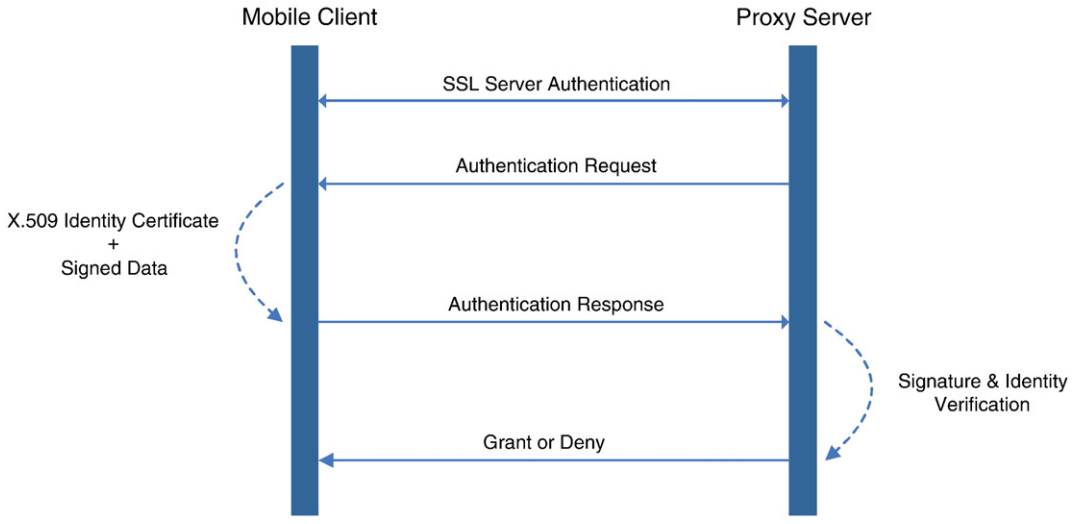


Fig. 7. Mobile client authentication.

GSI; therefore the mobile client signature verification, explained in last paragraph, could fail. Extended GSI integrity functions will be needed in order to implement the mobile client authentication protocol.

Trust is not provided by GSI. Thus, a trust framework might be needed. Furthermore, a protocol between the mobile and the proxy might be implemented in order to integrate mobile devices to this framework.

GSI provides functionality to manage identity components, such as X.509 identity credentials or proxy credentials. Nevertheless, extended GSI have to be able to handle access control lists used by the extended authentication service, and identity information used by the trust framework.

The authorization mechanisms included in GSI allow simple authorization scenarios. The Gridmap Authorization, provided by GSI, is based on the Gridmap file, which contains a list of distinguished names that are allowed access to a service. For more complex authorization scenarios, Globus Toolkit provides an infrastructure to deploy other authorization mechanisms. An authorization model, such as Open PMI, could be improved and deployed for the grid. The authorization data exchange protocol between the mobile device and the proxy will be implemented. This protocol will provide a way to integrate mobile security into the authorization mechanisms.

Privilege management is provided by Globus Toolkit for managing Gridmap authorization files, which is basically an access control list. The new authorization mechanisms integrated in the grid will need to operate with attribute certificates (which associate a subject to several attributes or privileges), and with information related to privileges, such as privilege delegation. These new authorization mechanisms can be fully integrated into the grid. Moreover, the trust management system will use the information provided by the privilege management in order to derive the trust values of the users.

The grid auditing infrastructure will provide grid users with mechanisms to define auditing policies. Extended GSI will provide classes to achieve these tasks and will integrate the new infrastructure into the GSI.

GSI offers secure storage, i.e., a mechanisms to prevent users to steal private data from other users. The files containing private data are encrypted using a password, and the data is recovered entering the password to decrypt the encrypted data. GSI uses this mechanism to store private keys. The extended GSI must implement classes to store security information in a safe mode. This type of information includes attribute certificates used by the authorization service, access control lists, etc.

Sometimes it is not necessary to store data in a safe mode. For example, GSI stores the private key of the proxy in a local storage system without being encrypted. Extended GSI must implement new classes to be able to deal with this type of storage, such as log files used by the auditing service, access control lists, and configuration files.

Credential delegation and single sign-on are dealt with by the GSI with the aid of proxy certificates. It is thus not necessary to extend any functionality of these services.

6. Future work and conclusions

This paper introduces the security framework that will be developed for the grid platform in the GREDIA project. We have followed the OGSA standards, and in order to design the security framework we have used the GSI, Grid Security Infrastructure. GSI already contains some of the most important security services that are needed for ensuring security in grid environments, but others must be implemented by the developers. In particular, we have paid special attention to authorization. Thus, we have mentioned the most important existing authorization models and have indicated which of these are most useful for the GREDIA framework. We believe that PERMIS and VOMS are the most suitable models in this context.

Aspects that are not included in the GSI but are needed in GREDIA include encryption and a security framework for mobile applications. We have highlighted the most important security features that need to be added to Application Development Platform, the core of the application execution of the GREDIA platform.

In the future work we intend to integrate PERMIS and VOMS within the GREDIA platform. We also intend to develop the encryption algorithms required by the multimedia files that will be used in the media application pilot of GREDIA.

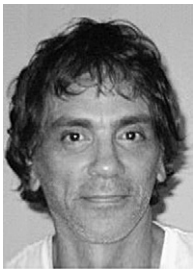
Acknowledgments

This work has been partially supported by the European Commission by the research project GREDIA (IST-FP6-034363) and the Spanish Ministry of Science and Education through the research project CRISIS (TIN2006-09242).

References

[1] R. Alfieri, R. Cecchini, V. Ciaschini, L. dell'Agnello, A. Frohner, A. Gianoli, K. Lorentey, F. Spataro. VOMS: An authorization system for virtual organizations. Proceedings of the 1st European Across Grids Conference, Santiago de Compostela, Feb. 2003.
[2] R. Anderson, R. Needham, A. Shamir, The steganographic file system, Proceedings of the Second International in Information Hiding, Portland, Oregon, USA, April 14–17, 1998, LNCS, vol. 1525, Springer.-Verlag, 1998, pp. 73–82.
[3] S. Androutsellis-Theotokis, D. Spinellis, A survey of peer-to-peer content distribution technologies, ACM Computing Surveys, vol. 36, 4, December 2004, pp. 335–371.
[4] V. Atluri. Security for Workflow Systems, Information Security Technical Report, Elsevier Science, vol. 6, 2, 2001, pp. 59–68.
[5] W. Bagga and R. Molva, Policy-based cryptography and applications, in 9th International Conference on Financial Cryptography and Data Security (FC'2005), 28 February–03 March 2005, Roseau, The Commonwealth of Dominica.
[6] S. Bajaj, et al., Web Services Policy 1.2 — Framework (WS-Policy) W3C Member Submission, 2006.
[7] Web Services Policy 1.2 — Attachment (WS-PolicyAttachment), W3C Member Submission, 2006.
[8] O. Berthold, H. Federrath, and S. Köpsell, Web Mixes: A system for Anonymous and Unobservable Internet Access. In Proceedings of Designing Privacy.
[9] B. Bhargava, C. Shi, S.Y. Wang, MPEG video encryption algorithms, Multimedia Tools and Applications, vol. 24, 2004.
[10] S. Boeyen, et al., Liberty trust models guidelines, in: J. Linn (Ed.), Liberty Alliance Project, Liberty Alliance, draft version 1.0, 2003.
[11] A. Bullock and S. Benford. An Access Control Framework for Multi-user Collaborative Environments. In *ACM GROUP*. Phoenix, AZ. 1999.
[12] D.W. Chadwick, A. Otenko, E. Ball. Role-Based Access Control with 470 X.509 Attribute Certificates. IEEE Internet Computing, vol. 7, No 2, March-April 2003.
[13] D. Chadwick. Authorization in Grid Computing. Information Security Technical Report, Elsevier, 10(1)33:40, 2005.
[14] D. Chaum, Untraceable Electronic Mail, Return Addresses and Pseudonyms. Com. ACM 24, 84–88.
[15] I. Clarke, O. Sandberg, B. Wiley, Freenet: a distributed anonymous information storage and retrieval system, Proceedings of the Workshop on Design Issues in Anonymity and Unobservability. Berkeley, CA, July 2000.
[16] M. Covington, W. Long, S. Srinivasan, A. Dey, M. Ahamad, G.D. Abowd. Securing context-aware applications using environment roles. In *ACM Symposium on Access Control Model and Technology*. Chantilly, VA. 2001.
[17] S.E. Czerwinski, et al., An architecture for a secure service discovery service, Proceedings of MobiCom '99, Seattle, WA, August 1999.
[18] GREDIA Integrated Architecture. www.gredia.eu.
[19] F. Dabek, M. Kaashoek, D. Karger, R. Morris, I. Stoica, Wide-area cooperative storage with CFS, Proceedings of the ACM (SOSP'01) Conference, Banff, Canada, 2001.
[20] http://eu-datagrid.web.cern.ch/eu-datagrid/.
[21] http://datatag.web.cern.ch/datatag/.
[22] R. Dingledine, M. Freedman, D. Molnar, The FreeHaven project: distributed anonymous storage service, Workshop on Design Issues in Anonymity and Unobservability, 2000, pp. 67–95.
[23] J. Douceur, The Sybil attack, Proceedings of the 1st International Workshop on Peer-to-Peer Systems (IPTPS'02), MIT Faculty Club, Cambridge, MA, 2002.
[24] P. Druschel and A. Rowstron, Past: A Large-Scale, Persistent Peer-to-Peer Storage Utility. In Proceedings of the Eight Workshop on Hot Topics in Operative Systems.
[25] www.earthsystemgrid.org.
[26] I. Foster, J. Geisler, W. Nickless, W. Smith, S. Tuecke, Software infrastructure for the I-WAY metacomputing experiment, Concurrency: Practice & Experience 10 (7) (1998) 567–581.
[27] I. Foster, C. Kesselman, S. Tuecke, The anatomy of the grid: enabling scalable virtual organizations, International Journal of High Performance Computing Applications 15 (3) (2001) 200–222.

[28] I. Foster, C. Kesselman, J. Nick, S. Tuecke, The physiology of the grid, Global Grid Forum, 2002.
[29] M. Freedman, E. Sit, J. Cates, and R. Morris, Introducing Tarzan: A Peer to Peer Anonymizing Network Layer. In Proceedings of the 1st International Workshop on Peer-to-Peer Systems (IPTPS'02).
[30] The Freedom anonymity system. Web site http://www.fredom.net.
[31] B. Furht, D. Socek, A.M. Eskicioglu, Fundamentals of multimedia encryption techniques, in: B. Furht, D. Kirovski (Eds.), Multimedia Security Handbook, CRC Press LLC, Boca Raton, Florida, 2004.
[32] http://www.globus.org/.
[33] C.K. Georgiadus, I. Mavridis, G. Pangalos, R. Thomas. Flexible Team-Based Access Control Using Contexts. In *ACM Symposium on Access Control Model and Technology*. Chantilly, VA. 2001.
[34] http://www.ggf.org.
[35] https://forge.gridforum.org/sf/projects/ogsa-authz/.
[36] T. Grandison, M. Sloman, A survey of trust in Internet applications, IEEE Communications Surveys, 2000.
[37] D. Goldschlag, M. Reed, P. Stevenson, Onion routing for anonymous and private Internet connections, Comm. ACM 42 (2) (1999) 39–41.
[38] www.Gridalliance.org.
[39] S. Hand, T. Roscoe, Mnemosyne: peer-to-peer steganographic storage, Proceedings of the 1st International Workshop on Peer-to-Peer Systems (IPTPS'02), MIT Faculty Club, Cambridge, MA, 2002.
[40] A. Harrison, C.D. Jensen, Cryptographic access control in a distributed file system, Proceedings of the eighth ACM Symposium on Access Control Models and Technologies, Italy, 2003.
[41] G. Herrmann, G. Pernul. Viewing business process security from different perspectives, in *11th International Bled Electronic Commerce Conference*, Slovenia, 1998.
[42] H. Humphrey, M.R. Thompson, K.R. Jackson, Security for grids, Lawrence Berkeley National Laboratory, Paper LBNL-54853, August 14 2005.
[43] R. Sheary, Internet Security Glossary, 2000 http://www.ietf.org/rfc/rfc2828.txt.
[44] A. Jøsang, R. Ismail, C. Boyd, A survey of trust and reputation systems for online service provision, Decision Support Systems 43 (2) (March 2007) 618–644.
[45] F. Kerschbaum, J. Haller, Y. Karabulut, P. Robinson, A trust-based reputation service for virtual organization formation, in: Ketil Stølen, William H. Winsborough, Fabio Martinelli, Fabio Massacci (Eds.), iTrust2006: Proceedings of the 4th International Conference on Trust Management, Lecture Notes in Computer Science, vol. 3986, 2006, pp. 193–205.
[46] J. Kubiatowicz, D. Bindel, Y. Chen, P. Eaton, D. Geels, S. Gumadi, H. Weatherspoon, W. Weimer, C. Wells, B. Zhao, Oceanstore: an architecture for global scale persistent storage, Proceedings of ACM ASPLOS, 2000.
[47] G. von Laszewski, B.E. Alunkal, I. Veljkovic, Towards reputable grids, Scalable Computing: Practice and Experience, vol. 6, 3, 2005, pp. 95–106.
[48] R. Lepro, Cardea: Dynamic access control in distributed systems, Technical Report NAS-03-020, NASA, 2003.
[49] X. Liu, A.M. Eskicioglu, Selective encryption of multimedia content in distribution networks: challenges and new directions, IASTED International Conference on Communications, Internet and Information Technology (CIIT 2003), Scottsdale, AZ, November 2003, pp. 17–19.
[50] T.Y. Li, H. Zhu, K.Y. Lam, A Novel Two-Level Trust Model for Grid. International Conference on Information and Communications Security, ICICS 2003. S. Qing, D. Gollmann, and J. Zhou (Eds), LNCS 2836, pp: 214–225.
[51] M. Lorch, D.B. Adams, D. Kafura, M.S.R. Koneni, A. Rathi, S. Shah, The PRIMA System for Privilege Management, Authorization and Enforcement in Grid Environments. In *Fourth International Workshop on Grid Computing*, 2003.
[52] M. Lorch, B. Cowles, R. Baker, L. Gommans, P. Madsen, A. McNab, L. Ramakrishnan, K. Sankar, D. Skow, M.R. Thompson. Conceptual Grid Authorization Framework and Classification, Global Grid Forum, 23 November 2004, www.gridforum.org/documents/GFD.38.pdf.
[53] D. Martin, et al., Bringing semantics to web services: the OWL-S approach, Proceedings of the 1st SWSWPC, USA, 2004.
[54] MojoNation. The MojoNation web site, http://www.mojonation.net.
[55] http://www.globus.org/toolkit/security/myproxy/.
[56] A. Nadalin, et al., OASIS WS-SecurityPolicy 1.2, December 4 2006.
[57] The Open Grid Services Architecture, Version 1.0, Global Grid Forum, 29 January 2005.
[58] Object Management Group (OMG), UML 2.0 Infrastructure Specification, November 7 2003. http://www.omg.org/docs/ptc/03-09-15.pdf.
[59] ITU/ISO Recommendation. X.509 Information Technology Open Systems Interconnection — The Directory: Autentication Frameworks, 2000. Technical Corrigendum.
[60] ITU-T Recommendation X.509. Information Technology Open Systems Interconnection - The Directory: Public-Key and Attribute Certificate Frameworks, 2000. ISO/IEC 9594-8:2001.
[61] L. Pearlman, V. Welch, I. Foster, C. Kesselman, S. Tuecke, A Community Authorization Service for Group Collaboration, *Proceedings of the 3rd International Workshop on Policies for Distributed Systems and Networks (POLICY'02)*, 2002.
[62] M. Pirretti, P. Traynor, P. McDaniel, B.r.e.n.t. Waters, Secure attribute-based systems, Proceedings of the 13th ACM Conference on Computer and Communications Security (CCS '06), Alexandria, Virginia, USA, 2006.
[63] T. Priebe, E.B. Fernandez, J.I., Mehlau, G. Pernul. A Pattern System for Access Control. Proc. *18th Annual IFIP WG 11.3 Working Conference on Data and Application Security*, Sitges, Spain, July 2004.
[64] M. Rabin, Efficient dispersal of information for security, load balancing and fault tolerance, J. ACM 36 (2nd April 1989) 335–348.
[65] S. Rhea, C. Wells, et al., Maintenance-free global storage, IEEE Internet Compu. (2001) 40–49.
[66] A. Sahai, B. Waters, Fuzzy identity-based encryption, advances in cryptology-Eurocrypt'05, LNCS 3494, Springer, 2005, pp. 457–473.
[67] R. Sandhu, P. Samarati, Access control: principles and practice, IEEE Communications 32 (9) (1994) 40–48.
[68] R. Sandhu, E.J. Coyne, H.L. Feinstein, C.E. Youman, Role-based access control models, IEEE Computer 29 (2) (Feb. 1996) 38–47.
[69] A. Serjantov, Anonymizing censorship resistant systems, Proceedings of the 1st International Workshop on Peer-to-Peer Systems (IPTPS'02), MIT Faculty Club, Cambridge, MA, 2002.
[70] http://www.shibboleth.internet2.edu.
[71] F. Siebenlist, V. Welch, S. Tuecke, I. Foster, N. Nagaratnam, P. Janson, J. Dayke and A. Nadalin. OGSA Security Roadmap, Global Grid Forum, Document 5, Open Grid Security Architecture Security Working Group, July 2003.
[72] B. Sotomayor, The Globus Toolkit 3 Programmer's Tutorial. Access Control with Gridmaps, http://www.casa-sotomayor.net/gt3-tutorial/multiplehtml/ch15.html.
[73] J. Treadwell, Open grid services architecture, Glossary of Terms, 2006. http://www.ogf.org/documents/GFD.81.pdf.
[74] R. Thomas, R. Sandhu. Task-based Authorization Controls (TBAC): Models for active and enterprise-oriented authorization management. *In Database Security XI: Status and Prospects*, T.Y. Lin, X. Qian, Eds. North-Holland. 1997.
[75] R. Thomas. Team-based access control (TMAC). In *Proceedings of 2nd ACM Workshop on Role-Based Access Control*. Fairfax, VA. 13–19. 1997.
[76] M.R. Thompson, A. Essiari, S. Mudumbai. Certificate-Based Authorization Policy in a PKI Environment. *ACM Transactions on Information and System Security (TISSEC)*, Volume 6, Issue 4 (Nov. 2003), pp 566-588.
[77] S. Trabelsi, J.C. Pazzaglia, Y. Roudier, Secure Web service discovery: overcoming challenges of ubiquitous computing, 4th IEEE European Conference on Web Services (ECOWS 2006), Zurich — Switzerland, December, 2006.
[78] S. Trabesi, Y. Roudier, J.C. Pazzaglia, Service Discovery: Reviewing Threats and Security Architectures, Research Report RR-07-197, Institute Eurecom, Mobile Communications Department.
[79] S. Trabelsi, L. Gomez, Y. Roudier, Context-aware security policy for the service discovery, 3rd IEEE International Symposium on Security in Networks and Distributed Systems (SNDS'07), Niagara Falls, Canada, May 21–23 2007.
[80] S. Tueche, et al., Internet X.509 public key infrastructure proxy certificate profile, IETF draft, 2001.
[81] Vollbrecht, P. Calhoun, S. Farrell, L. Gommans, G. Gross, B. de Bruijn, C. de Laat, M. Holdrege, D. Spence. AAA Authorization Framework, RFC 2904.
[82] M. Waldman, A.D. Rubin, and L.F. Cranor, Publius: A Robust, Tamper-Evident, Censorship-Resistant Web Publishing System. In Proceedings of the 9th USENIX Security Symposium.
[83] WS-Discovery Specifications, http://msdn.microsoft.com/ws/2005/04/ws-discovery/.
[84] http://www.uddi.org.
[85] www.oasis-open.org/committees/wsn.
[86] www.globus.org/wsrf.
[87] F. Zhu, M. Mutka, L. Ni, Facilitating secure ad-hoc service discovery in public environments, Proceedings of the 27th IEEE Computer Software and Applications Conference, USA, 2003.
[88] F. Zhu, M. Mutka, and L. Ni, Prudent exposure: A private and user centric.



Jose L. Vivas received his MS and PhD in Computer Science in 1996 and 2001, respectively, from the Royal School of Technology — KTH, Stockholm, Sweden. From May 2002 until December 2003 he worked as senior researcher at Hewlett-Packard Laboratories in Bristol, UK, and thereafter as researcher at the Computer Science Department of the University of Málaga, Spain. His research activities are focused on formal methods, security engineering, grid security, and security assurance.

Carmen Fernández-Gago is a postdoctoral researcher at the Department of Computer Science at the University of Malaga. She received her degree in Mathematics at the University of Malaga in 1996 and her PhD in Computer Science from the University of Liverpool (UK) in 2004. She also worked at this department as a postdoctoral researcher before taking up her current position. Her main research areas are Trust Management and Reputation System, mainly by using formal methods. Currently she is working actively on the European Commission funded projects GREDIA and SPIKE, from FP6 and FP7 respectively. She is also member of several program committees of international conferences and workshops as well as of a journal.

Javier Lopez received his MS in Computer Science and PhD in Computer Engineering in 1992 and 2000, respectively, from University of Malaga. After a period of four years as system analyst in the private sector, he joined the Computer Science Department at the same university, where he actually is Full Professor. Prof. Lopez has participated in research projects of the V, VI and VII Framework Programme, and in more than eighty program committees of international security and crypto events. Additionally, he is the Co-Editor in Chief of Springer's International Journal of Information Security, and member of the Editorial Boards of Computers & Security, Computers Communications, Security and Communications Networks Journal, International Journal on Critical Infrastructure Protection, Wireless Communications and Mobile Computing, and Computer Networks, among others. Additionally, Prof. Lopez is the Spanish representative of the IFIP TC-11 WG (Security and Protection in Information Systems).

Andres Benjumea received his Bachelor's degree in Computer Science in 1996 from the University of Malaga. From 1996 to 2005, he worked as a System Analyst. In 2006 he started working at the Computer Science Department at the University of Malaga. He has participated in several (national and international) security projects, including the OpenPMI authorization framework, and the European project GREDIA (GRid enabled access to rich mEDIA content).