Journal of Systems Architecture 57 (2011) 240–258

Contents lists available at ScienceDirect

Journal of Systems Architecture

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

Security services architecture for Secure Mobile Grid Systems

David G. Rosado [a,*], Eduardo Fernández-Medina [a], Javier López [b]

[a] University of Castilla-La Mancha, Alarcos Research Group-Information Systems and Technologies Institute, Information Systems and Technologies Department, ESI, Paseo de la Universidad 4, 13071 Ciudad Real, Spain
[b] University of Málaga, Computer Science Department, E.T.S. Ingenieria Informatica, Campus de Teatinos, 29071 Malaga, Spain

ARTICLE INFO

Article history:
Received 30 November 2009
Received in revised form 27 March 2010
Accepted 20 May 2010
Available online 1 June 2010

Keywords:
Security architecture
Security services
Mobile Grid Computing
Development process
V&V

ABSTRACT

Mobile Grid, is a full inheritor of the Grid with the additional feature that it supports mobile users and resources. Security is an important aspect in Grid based systems, and it is more complex to ensure this in a mobile platform owing to the limitations of resources in these devices. A Grid infrastructure that supports the participation of mobile nodes and incorporates security aspects will thus play a significant role in the development of Grid computing. The idea of developing software through systematic development processes to improve software quality is not new. However, many information systems such as those of Grid Computing are still not developed through methodologies which have been adapted to their most differentiating features. The lack of adequate development methods for this kind of systems in which security is taken into account has encouraged us to build a methodology to develop them, offering a detailed guide for their analysis, design and implementation. It is important to use software V&V techniques, according to IEEE Std. 1012 for Software Verification and Validation, to ensure that a software system meets the operational needs of the user. This ensures that the requirements for the system are correct, complete, and consistent, and that the life-cycle products correctly design and implement system requirements. This paper shows part of a development process that we are elaborating for the construction of information systems based on Grid Computing, which are highly dependent on mobile devices in which security plays a highly important role. In the design activity of the process, we design a security architecture which serves as a reference for any mobile Grid application that we wish to build since this security architecture defines a complete set of security services which will be instantiated depending on the requirements and features found in previous activities of the process. A V&V task is also defined in the design activity to validate and verify both the architecture built and the traceability of the artifacts generated in this activity. In this paper, we will present the service-oriented security architecture for Mobile Grid Systems which considers all possible security services that may be required for any mobile Grid application.

1. Introduction

The Grid is mainly focused on remote access to computational resources, thus solving the problem of coordinating the resources shared between virtual, multi-institutional and dynamic organizations [1]. Mobile Grid Computing is about making Grid Services available and accessible anytime anywhere from mobile devices. Wireless mobile devices are generally characterised by several limitations and constraints [2]. These limitations make mobile devices a platform that would benefit from the Grid. The main advantages of Mobile Grid Computing include mobile-to-mobile and mobile-to-desktop collaboration for resource sharing, improving user experience, convenience and contextual relevance and new application scenarios.

The Grid is an extensible, distributable, scalable service-oriented architecture (SOA) system [3]. SOA architecture for the Grid provides methods with which to expose services and allow computers to talk to each other in a highly heterogeneous environment. Services on the Grid may be individually purposeful (fully self-contained) or act as a component which may contribute a useful workflow. Services communicate with clients by exchanging the messages that define them, and also publish their capabilities. The benefits of an SOA architecture on the Grid is that it provides loose coupling which results in a system which is flexible, scalable, with services that are replaceable and with added fault tolerance [4]. A service-oriented security architecture for the Mobile Grid Systems has therefore been used and designed in our approach.

On the other hand, the growing need to construct secure systems, principally as a result of the new vulnerabilities caused by

* Corresponding author. Tel.: +34 926295300; fax: +34 926295354.
E-mail addresses: David.GRosado@uclm.es (D.G. Rosado), Eduardo.FdezMedina@uclm.es (E. Fernández-Medina), jlm@lcc.uma.es (J. López).

1383-7621/$ - see front matter © 2010 Elsevier B.V. All rights reserved.
doi:10.1016/j.sysarc.2010.05.009

the use of the Internet and the fact that the applications are distributed in heterogeneous environments, has encouraged the scientific community to demand a clear integration of security into the development processes [5–10]. The majority of existing Grid applications, that have clear differentiating features of which security is an extremely important aspect [11,12], have been built without the systematic development of a security process and are based on ad hoc developments [13,14]. The lack of adequate secure development methods for this kind of systems has encouraged us to build a development process with which to develop them, offering detailed guidelines towards their analysis, design and implementation [15].

The Verification and Validation (V&V) of models can be used to detect errors and unconsciousness in the earlier stages of the development, thus avoiding the transmission of these errors to the subsequent stages. V&V is a set of activities whose goal is to foster software quality during the development life-cycle. The IEEE Std. 1012 for Software Verification and Validation [16] is a recognized and utilized standard for V&V which has been used to define the V&V tasks that we have incorporated in our development process.

Our idea is to build a service-oriented security architecture and integrate it into the design activity of the development process that we are elaborating to develop Mobile Grid Systems in which we have integrated V&V tasks into the overall process, thus correcting the mistakes made during the software life-cycle which affect the quality of the final software product. In this paper we show a reference security architecture which allows us to design the specific security architecture of the application that we are developing. This security architecture has to be verified, indicating the fulfilment of the traceability of artefacts, i.e. whether the security architecture has been designed by considering the artefacts (mainly security use cases) generated in the analysis activity. This security architecture also has to be validated, indicating whether the designed architecture covers, fulfils and considers all the requirements specified in the analysis activity. This V&V task is defined in the development process (in the design activity) once the secure software architecture has been designed.

The main purpose of the process is to offer a secure development process to improve the quality and security of Mobile Grid Computing based systems. A preliminary version of the process has been presented in [15] in which we describe our general approach. An informal presentation of the first steps of this process has been provided in [17], which consists of analyzing the security requirements of Mobile Grid Systems directed by misuse cases and security use cases, and which is applied in an actual case study in [18] from which we obtain the security requirements for a specific application by following the steps described in our process. We have then gone onto elicit some common requirements of these kinds of systems, and these have been specified to be reused through a UML extension of use cases [19,20]. The new UML extension for Grid use cases can be used to build the main diagram of use cases for any Mobile Grid application, as is shown in [21]. These artifacts are used by the analysis activity of the process [22]. In this paper, we focus on the definition and design of the security service architecture which is used in the design activity of the process and will serve as a reference security architecture with which to build the specific security architecture of the different applications to be developed. This architecture must be verified and validated, detecting and correcting any possible mistakes made in the design and assuring that the security architecture fulfils its intended purpose and meets its users' needs.

This security architecture serves as a reference security architecture for any Mobile Grid System that we wish to develop since it defines a wide set of security services. Only a subset of these will be necessary for each Grid application to be developed, and will cover the specified security requirements for that application which are obtained in the analysis activity with the help of use cases diagrams. These use cases and their associated security requirements can be used to identify the set of security services of the reference security architecture through association rules between security requirements and security services.

The remainder of the paper is organized as follows: Section 2 presents related work with the security architectures and models for Grid systems. In Section 3 we briefly summarize the proposed process and provide a short definition of its activities. In Section 4, we propose the service-oriented security architecture for mobile Grid environments and we provide an in-depth description of all the security services of the architecture and the interface for each one of them. Section 5 shows the use of the security architecture in the development process, and finally, in Section 6, we put forward our conclusions and some research lines for our future work.

2. Related work

Existing Grid architecture and algorithms do not consider the mobile computing environment since mobile devices have not been seriously considered as valid computing resources or interfaces in Grid communities. Attention has recently been paid to the integration of these two emerging techniques of mobile and Grid computing, for example, in [23,24], although these works do not elaborate on how the mobile devices may be incorporated into the current Grid architecture.

OGSA [25] represents an evolution towards a Grid system architecture based on Web Service concepts and technologies which create new challenges in security and address specific Grid security requirements. The lack of support for mobile devices, including security aspects, makes this incomplete for Mobile Grid Systems. GSI [26] is an essential middleware component that has been integrated into many tools and offers solutions for Grid systems. These solutions are security solutions for Grid environments in which mobile devices are not considered, and this proposal does not offer solutions to the risks and possible attacks that appear in mobile computing.

EGEE Security [27] is a middleware architecture which supports security services for Grid environments. The authors define practice cases and tools which help to build the middleware and applications based on Grid computing, and which use Web service standards, but, like their predecessors, security aspects for mobile environments are not taken into account. EGA Security [28] defines a Grid management entity which manages a set of security functions and policies for Enterprise Grid to establish a secure connection with the Grid components that participate in the system. This approach does not consider mobile components in the enterprise Grid and consequently does not consider mobile security aspects.

Legion Security [29] defines a set of security mechanisms and policies to enable participants in a Grid system to expose their resources in a manner which is compliant with their local policies. This approach does not consider the security of mobile participants with mobile devices and a wireless network. Globe Security [30] constitutes a middleware level for a wide area distributed system. The security considerations presented in this approach for Grid systems therefore coincide with the security considerations for distributed systems, but are not exclusive to Grid environments. Mobile computing aspects are not taken into account. CRISIS Security [31] was developed to support wide area distributed applications and to define a wide set of security features which are present in many Grid systems, although they are not specific to this kind of systems. As with Globe, the mobile security aspects are not considered.

The approach presented in [32] proposes improving the security level of the Grid environment by integrating a new component into the security architecture that monitors the behavior of Grid applications, i.e. applications executed on behalf of Grid users, thus enhancing proposed fine grained run-time access control framework by integrating it with a powerful trust management framework, i.e., the Role-based Trust Management Language (RTML). Moreover, an implementation of a library for the management and evaluation of the Role-based trust management credentials and policies written in RTML, also extended by weights, in mobile devices has been presented in [33]. Although this approach is very interesting and provides solutions for trust management and credentials, our approach attempts to offer a global solution which considers a wide selection of security services to be incorporated within a security architecture for mobile Grid environments, and which is part of a complete methodological approach to develop this kind of systems.

Many proposals for Grid system security architectures and scalable approaches which use standards for their implementation also exist, but these offer no solutions to the incorporation of mobile devices or consider the security aspects that are so important in mobile computing. All are focused on security and the Grid tools and mechanisms necessary for its implementation. The security architecture which we are building is oriented towards services and is designed for a mobile environment, signifying that the security architecture offers security services, mechanisms, protocols and methods which fulfil the necessities and requirements of Grid Systems and Mobile Computing.

3. Overview

The structure of the process which we propose follows the classical cycle, in which we find a planning phase, a development phase including analysis, design and construction and finally a maintenance phase. However, the proposed process is specially designed for this kind of systems since their particular aspects and features are considered in each of the activities of the process. Moreover, this process involves software verification and validation (V&V) activities to mitigate certain software development risks throughout the entire software development life-cycle. In particular, we define V&V tasks in the analysis, design and construction activities which validate and verify the output artifacts and model of each activity in the process. Further details of the activities and tasks in our process can be found in [15,17,18]. Fig. 1 shows the structure of the process using SPEM (Software & Systems Process Engineering Metamodel) version 2.0 [34].

The planning phase has only one activity: "Secure Mobile Grid System Planning", in which an initial capture of requirements and necessities should be carried out in order to elaborate a development plan.

The development phase is composed of three activities: analysis, design and construction.

- The "Secure Mobile Grid System Analysis" activity is centred on identifying and analyzing the requirements and security requirements of Grid systems from a reusable use cases model in which the use case diagrams and security use cases for this kind of systems are defined. In this activity a V&V task is defined to verify whether the analysis artifacts have been correctly generated, and whether the different UML diagrams (built according to the UML profile) are related and coordinated. This task validates whether the UML diagrams correctly define and describe the behavior of the different scenarios with the purpose of identifying the requirements in conflict and ambiguous requirements. These requirements can be analyzed and corrected in order to improve all the artifacts in the following iterations of this activity before continuing with the design activity. This V&V task attempts to establish an analogy with the activities of "Concept V&V" and "Requirements V&V" of the IEEE Std. 1012 which verify the allocation of system requirements, validate the selected solution, and ensure that no false assumptions have been incorporated in the solution, and ensure the correctness, completeness, accuracy, testability, and consistency of the system software requirements and of the security requirements.

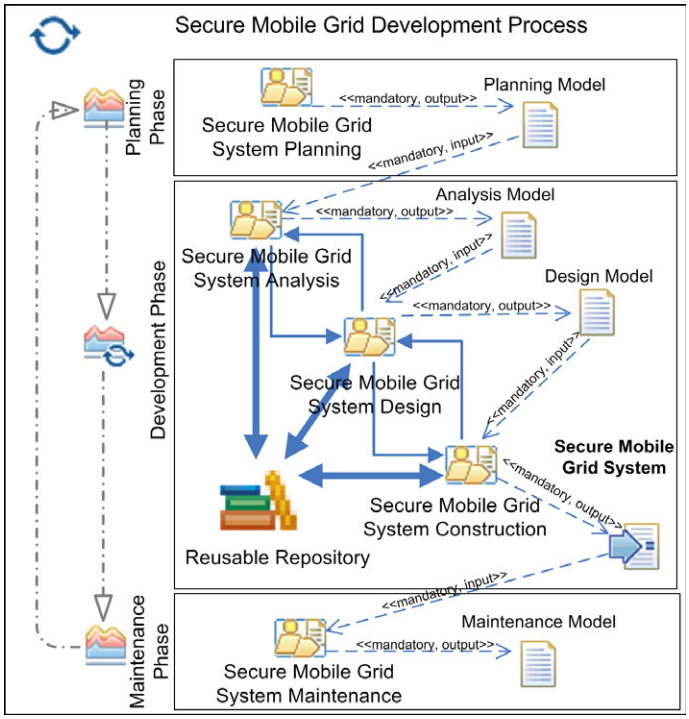


Fig. 1. Structure of our process.

- The goal of the "Secure Mobile Grid System Design" activity is to define a secure software architecture in which we should select the structural elements and the security structural elements of which the system is composed, and the behavior and interfaces between them. Before proceeding with the next activity, it is necessary to verify and validate the secure software architecture through a specific V&V task in which we check that the architecture designed is correct and valid and that the artifacts have been correctly generated. The architecture obtained in this activity has to be validated with the requirements specified in the analysis activity and the traceability between artifacts has to be verified, and the possible conflicts or errors in the design must be identified and analyzed for their subsequent refinements in subsequent iterations of this activity. Moreover, it verifies that the other scenarios are fulfilled with the architecture design, identifying possible conflicts, studying their scope and analyzing the opportunities to correct and deal with them. Therefore, this task has been defined in three steps: (a) validate designed architecture; (b) verify design model; and (c) identify, study and analyze the conflicts. The input artifacts for this task are: the analysis model generated in the previous activity and security software architecture built in the previous task of this activity. This V&V task obtains the output artefact such as the design conflicts of the current iteration found in the realization of this design activity after carrying out the aforementioned steps from the input artifacts. This V&V task is related to the "Design V&V" activity of the IEEE Std. 1012 which demonstrates that the design is a correct, accurate, and complete transformation of the software requirements and security requirements and that no unintended features are introduced, analyzing relationships, evaluating the design elements, and verifying and validating that the software design interfaces and the services for correctness, consistency, accuracy, readability, and testability. Moreover, this task verifies that logic design and associated data elements correctly implement the critical requirements and introduce no new hazards, and that the architecture and detailed design outputs adequately address the identified security requirements.
- In the "Secure Mobile Grid System Construction" activity, the implementation model (components and deployment diagrams) are refined, and a Grid technological platform should be selected to build the design model obtained in the last activity, and to implement and test the secure software architecture, defining security services together with security mechanisms and protocols for our security architecture. This activity defines a V&V task which validates that the implemented system covers and considers all the requirements and aspects identified and analyzed for the application (in the analysis activity), especially security. It also verifies that the system has been implemented by following the design model and that the traceability of artifacts is correct. If any conflict or error is found during the running of this task, these conflicts or errors must be indicated for their future correction in subsequent iterations of the activity. The output artifacts in the V&V task are a list of construction conflicts of the current iteration, indicating the conflicts, errors, invalid installation and configuration, etc. found, which will be refined and solved in the following iteration of the task. The aim of this activity is to transform the system design into code, database structures, interfaces, services, etc. The objective of this V&V task is to verify and validate that these transformations are correct, accurate, and complete according to the "Implementation V&V" activity of the IEEE Std. 1012.

The maintenance phase has only one activity: "Secure Mobile Grid System Maintenance" and this is a typical maintenance activity in any development process, in which a maintenance plan of the system is defined according to the new necessities of the client, for its later modification.

Traceability is a property of the process in which the artefacts generated in an activity are used as input artefacts for the following activity where new artefacts are generated by considering the artefacts from the previous activity. Each of the activities in the process contain a V&V task to check for mistakes and to ensure that the final artefacts of each activity satisfy system requirements and are generated from the artefacts of the previous activity.

4. Security architecture

A security architecture describes how the system is put together, identifying the appropriate security mechanisms needed to satisfy the security requirements in order to protect the system from threats. We have elaborated a service-oriented security architecture in which we have defined a complete set of security services and interfaces which cover the majority of security requirements specified in the analysis activity for Mobile Grid Systems. This security architecture protects the Grid system and offers the support necessary to ensure that the security requirements and needs are fulfilled. This architecture must be integrated with the software architecture of the Grid system, contributing with security aspects and converting software architecture into the secure software architecture that will subsequently be implemented. The set of security services that will take part in the security architecture are shown in Fig. 2, in which service levels from the basic services to advanced services are represented.

These are security services from the reference security architecture which cover the full range of security needs and requirements that may appear in any Mobile Grid System. These services are related to each other, indicating what one service needs from another to carry out its function. This architecture is defined by levels, from the most basic to the most advanced, and all the security services can be related to each other, independently of the level to which they belong.

The different relationships between services are shown in Fig. 3. The arrows establish relationships between one service and another, indicating that one service can invoke operations from the interface defined by another service (direction of the arrow). So for example, the confidentiality service can invoke operations from the interface of the integrity service, and the operations from the interface of the confidentiality service can be invoked by the privacy service. Moreover, all security services invoke the operations of the interfaces of the policy services (Grid Security Policy Service and Mobile Policy Service) that are responsible for managing all security policies associated with each service.

Each service will have an interface which represents the door of communication with that service, so that it can interact with other services within the architecture, and with services that are external to the architecture. The interface defines a set of operations which perform a specific function within the service, offering results to those who invoke them. This is the means of communication between the various services in the architecture. One security service communicates and relates to another, through calls to its operations, obtaining results which depend on the input parameters to the operation passed by the service which invokes it. The interfaces of the security services are defined in two ways:

(i) specifically, in order to focus on and identify the objective that we wish to attain;

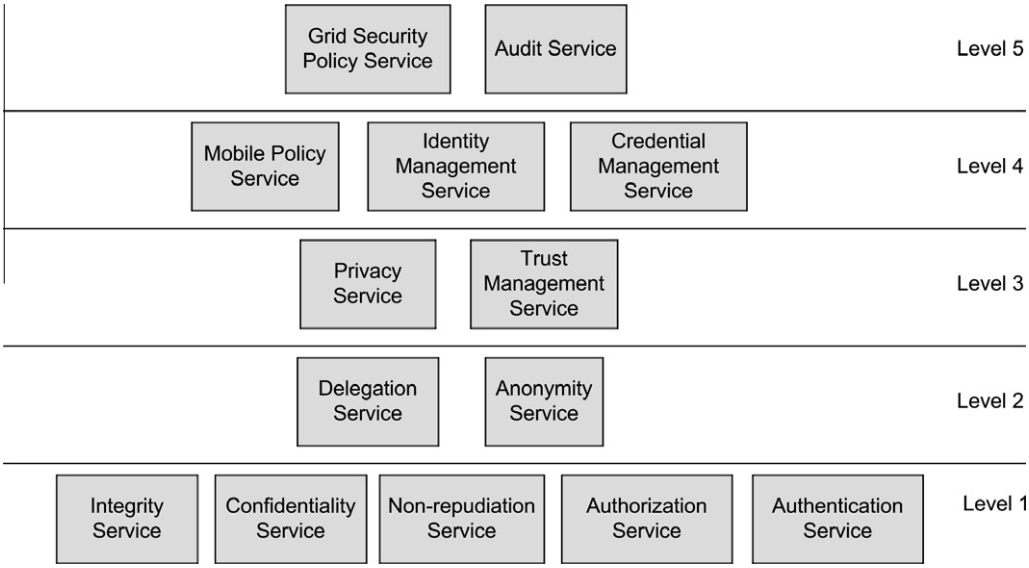


Fig. 2. Levels of security services of the proposed architecture.

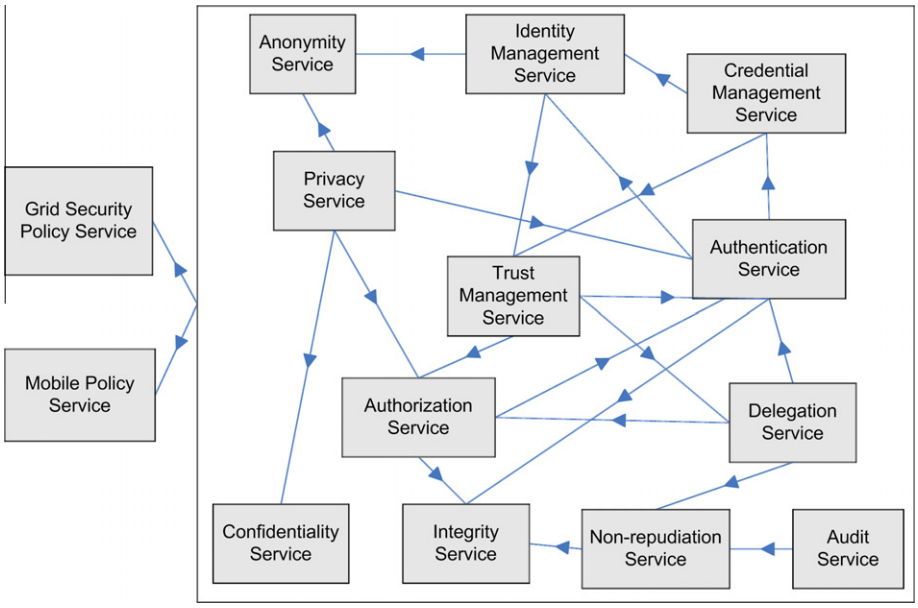


Fig. 3. Relationships between security services.

(ii) in a sufficiently generic manner, so as not to depend on any technological platform at the design level. We can therefore choose and adapt the appropriate implementation of each service with a specific technology in the implementation activity.

An in-depth description of each of the security services in the security architecture for Mobile Grid environments is shown as follows. We will describe the most important aspects of the service, the operations that make up the interface of each service, and the relationships of each service with the rest of services in the security architecture.

4.1. Authentication service

Authentication refers to the process of confirming or denying an individual's claimed identity. Authentication mechanisms are based on one or more of the following three classes of procedure [35]: proof by knowledge, proof by possession, and proof by property. Effective authentication solutions are a reliable way of permitting only authorized users to access a network.

Authentication in wireless networks is a challenging task. In a wireless environment, the radio medium can be accessed by anyone who has the appropriate wireless equipment. In such an "open" environment, authentication of communicating wireless devices becomes crucial for the secure exchange of sensitive data. Different kinds of wireless networks pose unique security issues in the design of authentication protocols [36]. Several methods can be used to authenticate services. Techniques include: passwords, one time pass and certificates. Password-based authentication must use strong passwords. Password authentication alone may be insufficient. It may be necessary to use vulnerability assessment to combine password authentication with other authentication and authorization process such as certificates, Lightweight

Directory Access Protocol (LDAP), Remote Authentication Dial-in User Service (RADIUS), Kerberos, and Public Key Infrastructure (PKI).

4.1.1. Authentication Interface

A principal must establish its credentials before it can invoke a service securely. If a user requires authentication and has not been authenticated prior to calling the Grid service, the (login) client must invoke the Authentication service to authenticate him/herself and then select optional attributes. The client presents his/her credentials to the service to be authenticated, and the authentication service authenticates these credentials (identity, attributes, etc.).

The user provides the authentication service with identity and authentication data (such as a password), and the service then authenticates the principal (in this case, the mobile user) and obtains credentials for it containing authenticated identity and privileges. A credential holds the security attributes of a principal. These security attributes include its authenticated (or unauthenticated) identities and privileges and information for establishing security associations. It provides operations to obtain and set security attributes of the principal it represents. We must consider the authentication policies associated with the service and mobile policies of the mobile devices involved in carrying out the exchange of information in a manner consistent with the correct protocols, mechanisms and formats for its understanding.

This interface defines two operations, one of which is in charge of authenticating the user or service identity by using the credentials passed as arguments to the method, and another which verifies the authenticity of the entities involved in a given context. This interface is related to the Credential Management Service which checks the validity of the credentials and extracts the necessary information for their authentication, and to the Grid Security Policy Service and Mobile Policy Service in order to obtain the authentication policies of both the Grid system and mobile devices.

- The **Credential authenticate**(**Credential**) method confirms whether the mobile user is the individual that s/he claims to be. The operation returns the authenticated credential that will be returned to the applicant or stored in a repository for subsequent use in various services that require it.
- The **Boolean isAuthenticated**(**context**) method confirms whether the identities of the entities involved in a specific context have been authenticated. In a communication in the Grid system, everyone involved (mobile devices, services, resources, users, etc.) should be authenticated to perform certain actions, signifying that entities which are within a context, for example, in a message request, must be authenticated if the action is to be performed. This operation is used in the authorization service to verify that the entities involved have been authenticated before performing the access request.

4.1.2. Relationships with other services

The Authentication Service establishes relationships with many other security services (see Fig. 4). These services call up the interface methods to authenticate the identity and credentials of a user. Moreover, the authentication service can call up interface methods of other services, offering specific functions which are necessary to execute the authentication function. For example, it can invoke the "*getIdentity()*" method of the Identity Interface to extract the identity of a certificate, or the Delegation service can invoke "*isAuthenticated()*" of this service to ensure that the credentials to be delegated are authenticated.

4.2. Authorization service

Authorization refers to mechanisms that decide when a user is authorized to perform a certain task. Authorization is related to authentication because, before a decision is made as to whether a user can (or cannot) perform a certain task (authorization), it must be clear that the user is who s/he claims to be (authentication). The core problem with authorization in a Grid setting is how to handle the overlay of policies and other assertions from

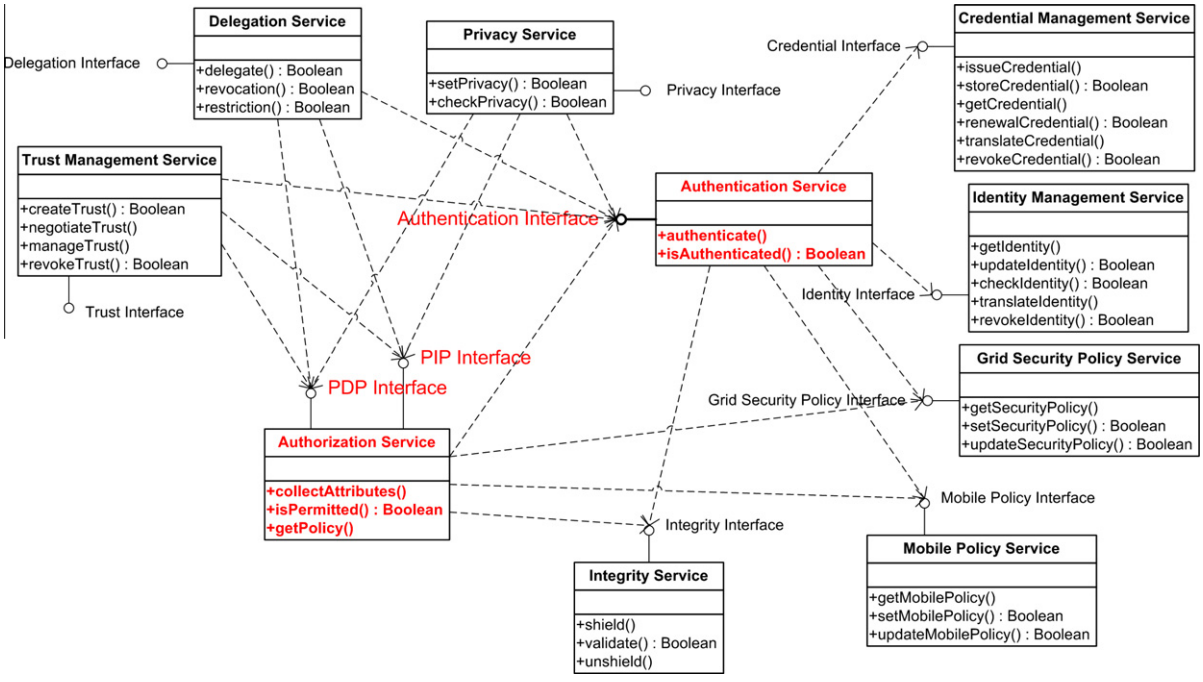


Fig. 4. Relationships between the authentication service, authorization service and the remaining services of the architecture.

multiple administrative domains (user policy, VO specific policy, operational procedures, site-local policy).

The authorization service of the architecture for mobile Grid environments is based on the XACML authorization architecture which consists of several logical components. A reference monitor concept is first used to intercept access requests. This component is called a Policy Enforcement Point (PEP). A PEP transmits access requests to a Policy Decision Point (PDP) for the retrieval and evaluation of applicable policies. Policies are specified and stored in Policy Administration Points (PAPs). If a PDP needs attributes of subjects, objects, or the environment that are missing in the original request, Policy Information Points (PIPs) deliver the data needed for evaluation. The access decision given by the PDP is sent to the PEP. The PEP fulfils the obligations and either permits or denies the access request according to the decision of the PDP. A PDP and a PEP may be co-located or separate depending on the system architecture. When co-located, the necessary communication is usually provided by a programming interface; when in separate resources, a secure communication mechanism and a request–response protocol is needed.

4.2.1. Authorization interfaces

The main components of any authorization service are the PDP to make decisions in accordance with certain attributes, and the PIP to obtain the necessary attributes for the decision of the PDP. If we define PDP and PIP as interfaces of the authorization service, we are defining this service in a general manner which is open to any Grid technology that is able to implement the operations defined in these interfaces using any programming language. The technology is used to associate the implementations of PDPs and PIPs in the security service in order to decide whether to allow or deny access to a resource or service Grid.

A definition of the operations of these interfaces and a brief description of each of them is shown below.

4.2.1.1. PIP interface. This interface defines a single operation that is responsible for obtaining the attributes required by the authorization service to make decisions. The attributes are obtained from four different types of entities: requestor, resource, action and environment.

- The **collectAttributes(parameters)** method is used to collect attributes that are relevant to making the authorization decision. The parameters should define at least one authenticated *subject* for which attributes should be collected, the *context* which holds properties of this XML message exchange, and the *operation* that the subject wishes to invoke. Therefore, the requestor or resource attributes are the attributes of the subject parameter, the action attributes are the attributes of operation parameter, and the environment attributes are the attributes of context parameter. The context parameter may have more attributes related to the subject, target or action invoked.

4.2.1.2. PDP interface. This interface defines the operations that are responsible for making the decision to allow or deny access, taking into account three entities: the subject who initiates the request (usually a user), the action to be performed and the context in which this occurs (generally an XML message). The decision will be made by taking into account the attributes of the entities involved and the security policy associated with the PDP.

The operations are, therefore:

- The **Boolean isPermitted(parameters)** method, which indicates whether the corresponding mobile subject/user is permitted to perform an action or access a Grid resource. This should return true if the local policy allows the subject to invoke the operation, or false otherwise. The parameters should define at least an authenticated client *subject* with credentials and attributes, the *context* which holds properties of this XML message exchange, and the *operation* that the subject wishes to invoke.
- The **Policy getPolicy(identifier)** method, which returns the security policy associated with the PDP used in the authorization (identifier). This method communicates with Grid Security Policy or Mobile Policy Services to obtain the security policies required.

4.2.2. Relationships with other services

The Authorization Service establishes relationships with many other security services (see Fig. 4) which call up the methods of the interface to request authorization about any action to be carried out. This service can also call up the interface methods of other services which are necessary to achieve the authorization goals. For example, the decision as to whether a user has the correct permissions to execute an action in a mobile device is based on the associated service of user domain and device domain policies, so it must first obtain the associated policies (by invoking "*getSecurityPolicy()*" or "*getMobilePolicy()*" from the Grid security policy interface and mobile policy interface, respectively). Moreover, if an entity wishes to delegate its credentials to a third party, the delegation service checks whether the delegator entity has authorization to carry out this action (by invoking "*isPermitted()*" from the authorization interface).

4.3. Confidentiality service

Confidentiality ensures that information is accessible only to those with authorized access to it. Confidentiality prevents eavesdropping. A malicious attacker could eavesdrop on wireless communications and compromise sensitive data, such as passwords. Confidentiality can be achieved by establishing an encrypted tunnel between the roaming stations and a portal on the wired infrastructure.

The WTLS protocol uses digital certificates to create a secure, confidential communications "pipe" between two entities, typically a mobile phone and a WAP Server. Data transmitted over a WTLS connection cannot be tampered with or forged without the two parties becoming immediately aware of the tampering. WTLS generally uses RSA-based cryptography. However, the protocol can also use elliptic-curve cryptography (ECC), which provides a high level of security while demanding fewer computing and memory resources than other encryption approaches. This is an important consideration for small-footprint handheld devices.

4.3.1. Confidentiality interface

Confidentiality is established by using cryptographic algorithms that provide the data with a certain amount of protection. The confidentiality service will therefore consist of protecting data by using encryption techniques, and removing the protection with decryption techniques. This interface defines two operations: *hide* to protect data and *reveal* to extract the information from the data protected.

- The **Data hide(parameters)** operation applies confidentiality protection to data. The *parameters* contain: (i) the data that we wish to protect; (ii) the keys used (public key, symmetric key, etc.) for this protection; and optionally, (iii) the mechanism or mechanisms used to implement this protection, if in the privacy policy associated with the service is not defined or defines something different to the that which we desire for these data. The operation returns the new data protected by encryption mechanisms.

- The **Data reveal(parameters)** operation removes the protection from a previous hide operation afforded to the data. The parameters should indicate the data that is protected and which must be deciphered, the keys used for encryption, and, optionally, the mechanism used for protection. This operation returns the original data before applying the confidentiality.

4.3.2. Relationships with other services

The Confidentiality Service establishes relationships with many other security services (see Fig. 5). The Privacy service can call up the interface methods to ensure the confidentiality of data or messages. This service can also call up the interface methods of other services, such as integrity interface, Grid security policy interface and mobile policy interface. These calls add the integrity property to the confidential data or messages, and they obtain the security policies associated with the confidentiality service and with the domains involved.

4.4. Integrity service

The integrity service ensures that messages (data) communications are not tampered with while in transit or in storage (in the device's memory, for example). A secure communication must ensure the integrity of the transmitted messages (data). This means that the receiving end must be able to know for sure that the message (data) s/he is receiving is exactly the same as that which the transmitting end has sent. Integrity ensures the correctness or accuracy of data, and that the data is protected against unauthorized modification, deletion, creation, and replication. Integrity features might also indicate unauthorized activities. 802.11 technology uses an Integrity Check Value (ICV) field in the packet. ICV is another name for message integrity check (MIC). WEP uses CRC-32 as an integrity check. The bluetooth standard relies on a cyclic redundancy check (CRC) to provide message integrity.

4.4.1. Integrity interface

This service defines a set of operations which support the integrity of messages or data in the system. An interface of the integrity service needs to define an operation to protect the integrity of the data or messages being passed as parameters and uses a particular mechanism for integrity. It also defines a recovery operation of the original data from the data protected by performing the inverse operation to that of protection, and an operation to validate and verify the integrity of data or messages at a given moment. The operations defined in this interface are:

- The **Data shield(parameters)** operation, which applies integrity protection to data. The data that we wish to protect and the keys used (public key, symmetric key, etc.) for their protection are indicated in *parameters*. The option of defining the mechanism or mechanisms used to implement this protection, in the case that the integrity policy associated with the service is not indicated or that we wish to apply a specific mechanism for this data in mobile computing, also exists. The operation returns the data protected.
- The **Boolean validate(Data)** operation, which checks integrity-protected data for modifications. The operation checks whether the data passed as a parameter complies with the previously established integrity property and has not been altered or tampered with.
- The **Data unshield(parameters)** operation, which converts integrity-protected data into the data originally shielded. The parameters must indicate the data protected, which contains the original data, the keys used for the establishment of the integrity, and (optionally) the mechanism used for the protection. The operation returns the original data before applying the integrity.

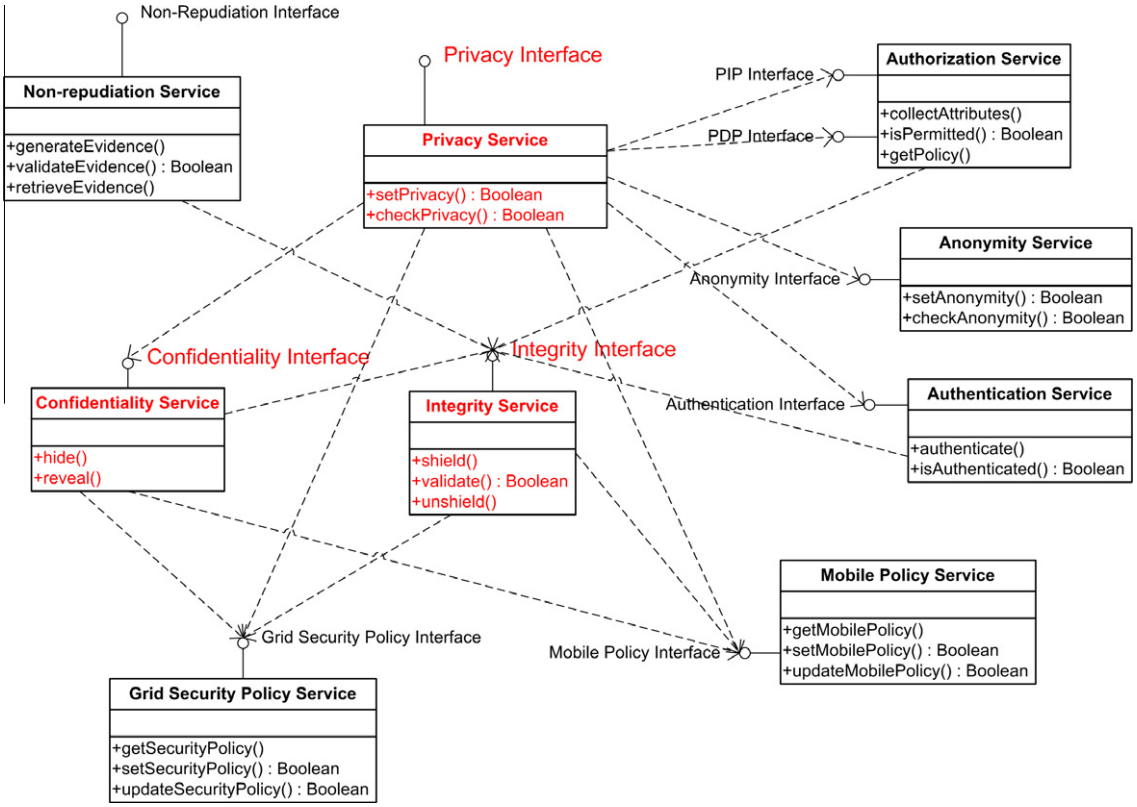


Fig. 5. Relationships between the confidentiality service, integrity service, privacy service and the remaining services of the architecture.

4.4.2. Relationships with other services

The Integrity Service establishes relationships with many other security services (see Fig. 5). Its interface methods are invoked by other security services (and Grid external services) such as authentication, authorization, confidentiality and non-repudiation services which wish to establish integrity in all the communications and data that are carried out and exchanged in these services. It also has relationships with the mobile policy interface and the Grid security policy interface to call up the methods in order to manage and obtain security policies associated with the service.

4.5. Non-repudiation service

The non-repudiation service collects, maintains, makes available and validates irrefutable evidence concerning a claimed event or action in order to resolve disputes regarding the occurrence or non-occurrence of the event or action.

The non-repudiation service may be provided through the use of mechanisms such as digital signatures, encipherment, notarization and data integrity mechanisms, with support from other services such as time stamping. Both symmetric and asymmetric cryptographic algorithms can be used for non-repudiation. The non-repudiation service can use a combination of these mechanisms and services, as appropriate, to satisfy the security requirements of the application in question.

4.5.1. Non-repudiation interface

The non-repudiation service in this specification provides generation of evidence of actions and later verification of this evidence, to prove that the action has occurred. Data is often associated with the action, so the service needs to provide evidence of the data used, along with the type of action.

An application can generate evidence associated with an action so that it cannot be repudiated at a later date. All evidence and related information is carried in non-repudiation tokens. Depending on the underlying cryptographic techniques used, the evidence is generated as:

- A secure envelope of data based on symmetric cryptographic algorithms requiring what is termed as a trusted third party (ttp) as the evidence generating authority.
- A digital signature of data based on asymmetric cryptographic algorithms which is assured by public key certificates, issued by a Certification Authority.

This service necessitates operations with which to: generate evidence of a specific event or action that has occurred in the system; validate the existing evidence, verifying any information (originator, recipient, date&time, etc.); and retrieve a particular evidence of a given event.

- The **generateEvidence(parameters)** operation generates evidence for an event or action. The parameters must indicate the originator and recipient of the action or event, the data sent or received or other information involved (keys, digital fingerprint, etc.), date and time when the action took place and a trusted third party, if it is used, according to techniques used. The security policy associated with this service specifies the most appropriate steps and techniques to be used to generate evidence.
- The **Boolean validateEvidence(parameters)** operation validates and verifies a piece of evidence. The parameters must indicate the evidence to be validated (generated previously), the trusted third party to verify that this was the authority that generated or signed the evidence, the subjects involved which must be valid subjects, and the action that initiated the evidence. The security policy indicates which parameters are needed to validate a piece of evidence. The operation returns true if the evidence is valid, or false otherwise.
- The **retrieveEvidence(parameters)** operation retrieves evidence related to a specific event or action. The parameters indicate an identifier of the evidence to be retrieved, or an action to recover all the evidence associated with this action or the date on which the action occurred, or the originator that generated it, and so on.

4.5.2. Relationships with other services

The non-repudiation service establishes relationships with other security services (see Fig. 6). This service relates to an integrity service in order to establish data integrity in the data and logs generated by the service itself. It also relates the Mobile Policy Service and the Grid Security Policy Service to obtain and manage the security policies of both the Grid and mobile resources. Other services can make use of the non-repudiation interface methods to manage evidence of what is happening in the system. Therefore, once the delegation service has delegated credentials, it must generate evidence of this delegation, and the audit service that uses the methods to complete the audit work.

4.6. Delegation service

Services that are used transiently by a user may need to be able to perform actions on a user's behalf without the user's direct intervention. For example, a computational job running overnight might need to access data stored in a different resource. Since there may be no direct trust relationship between the VO in which the service is running and the VO in which it wishes to make a request, the service needs to be able to delegate authority to act on the user's behalf.

The main problem in delegation is that when authority is passed from one entity to another, it is possible that the delegated credentials may be misused. These risks are minimized by giving the delegated credentials a limited lifetime.

4.6.1. Delegation interface

The delegation service must have a set of operations that will enable the delegation of credentials between entities, the possible revocation of delegated credentials, and the ability to restrict initially delegated rights at any given time of the lifetime of the credential. This minimum set of operations is defined as follows:

- The **Boolean delegate(parameters)** method, which permits the access rights of an entity to be propagated to a set of trusted entities, without explicitly changing its policy or requirements. The credentials that we wish to delegate (including attributes and rights), the destination to which they are delegated, and a limited lifetime are indicated in the *parameters*. The delegated entity may need to delegate another entity to perform the function required to execute the required function. To do this, the delegator entity must make this option available to the delegated entity so that it can delegate to other entities in a cascade. If the delegation is successful, true is returned.
- The **Boolean revocation(Credential)** method, which allows the delegator to cancel delegations it has issued (whether policy-based or delegation-based). A credential or credential identifier is passed as a parameter and the service takes charge of cancelling and excluding it from the system. If the credential is cancelled, the operation returns True.

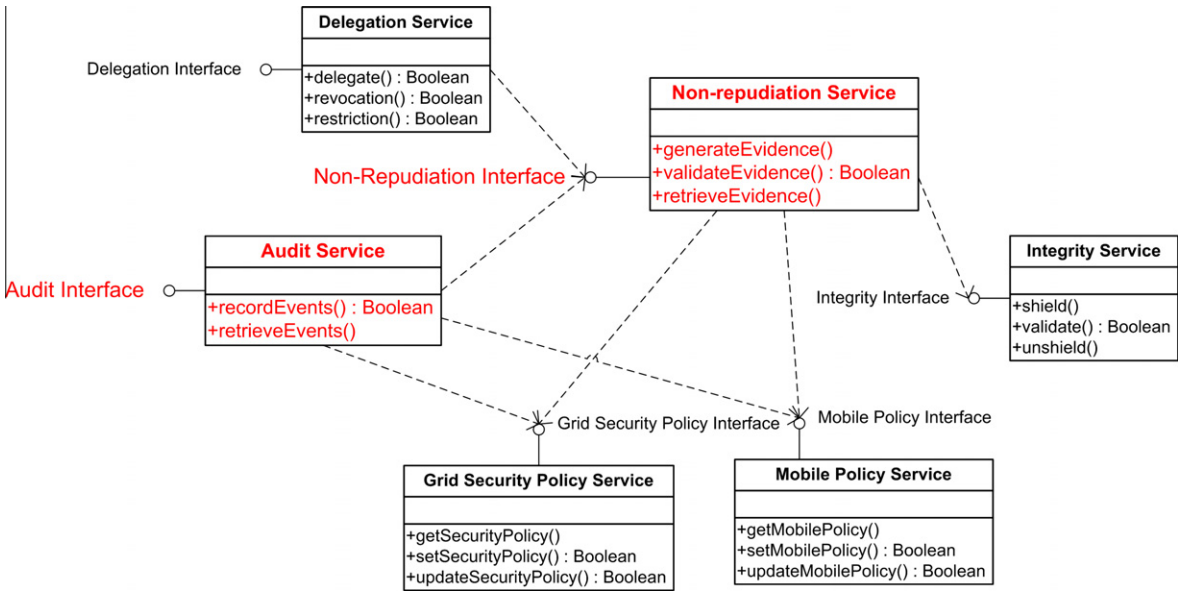


Fig. 6. Relationships between the non-repudiation service, audit service and the remaining services of the architecture.

- The **Boolean restriction(parameters)** method, which allows the delegator to be able to limit the rights granted by any delegation. We must indicate the delegate credential, the new values for the rights and attributes, and the delegated entity that contains and manages the delegated credential. The delegate credential uses this to modify the rights originally granted to the entity delegated. If everything is successful, it returns true.

4.6.2. Relationships with other services

The delegation service establishes relationships with many other security services (see Fig. 7) to check the identities of those involved in the delegation (relation with authentication service invoking isAuthenticated()), to request permission to carry out (relation with authorization service invoking isPermitted()) and to record the action (relation with the non-repudiation service invoking generateEvidence()). It also invokes mobile policy interface and Grid security policy interface methods which manage and obtain security policies associated with the service. The Trust service can use this service to handle the established delegations and the trust level of the involved entities and domains in the delegation.

4.7. Anonymity service

Anonymity requires that a user should have total control over the personal information disclosed by his/her home domain when

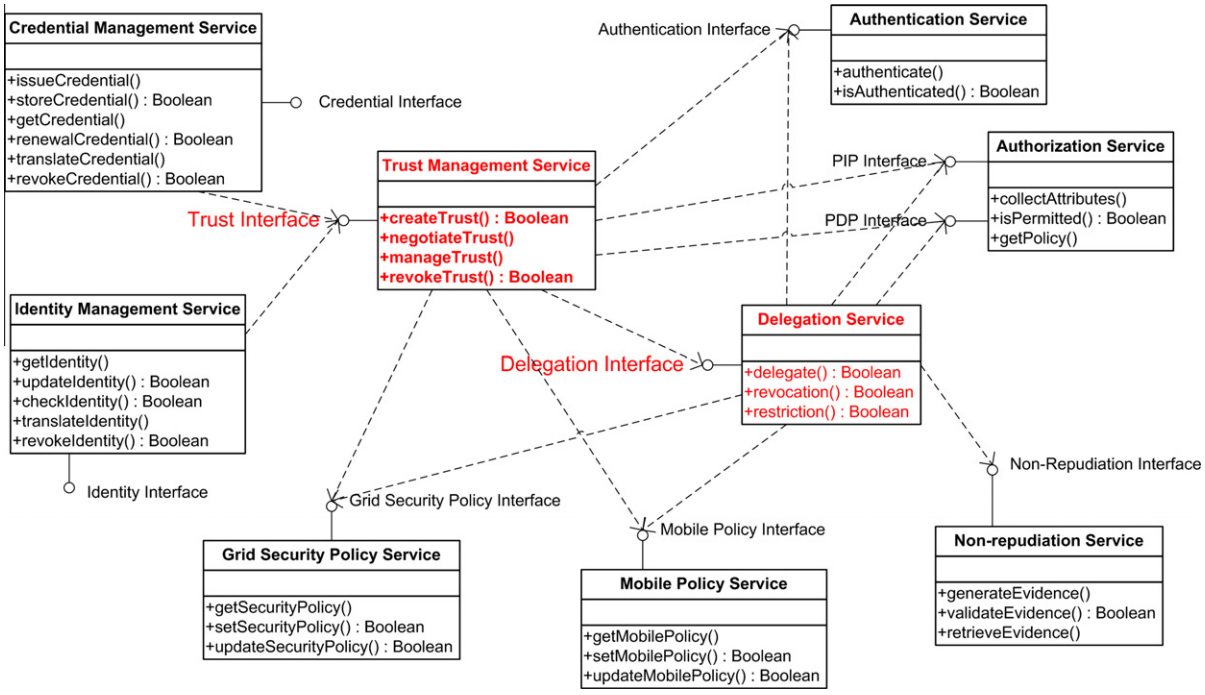


Fig. 7. Relationships between the delegation service, trust management service and the remaining services of the architecture.

the user accesses other service providers' services. Information about a particular person or organization is private and should only be known to its owner and to whoever s/he grants access rights. Privacy should be preserved in any kind of information system, be it fixed or mobile. The type of information that users may wish to keep private could include their real user identity when on-line, their activities, their current location and their movement patterns.

Preserving anonymity [37] is of greater concern in mobile systems for several reasons. Mobile systems yield more easily to eavesdropping and tapping, in comparison to fixed networks, making it easier to tap into communication channels and obtain user information. Current network implementers of mobile communication systems store a lot of user related information on network databases, especially for mobile telecommunication networks. This is done to assist in user mobility support, billing and authentication. This makes the user information more widespread and highly available. It is also uncertain whether the environment in which this data is stored is safe and trustworthy [38].

4.7.1. Anonymity interface

This service provides anonymity to mobile users involved in the Grid system, protecting data and any information related to the user, thus avoiding its exposition and publication by external parties. The mobility of mobile users between networks and domains signifies that it is possible to share identity information, authentication, etc. among the various domains to which is related. It is essential that mobile users have anonymity so that their data cannot be viewed or manipulated by the various domains with different security levels. The interface of the service should provide a method for establishing anonymity for a subject with regard to fulfilled actions, and an operation to verify that the ownership of anonymity is preserved during the lifetime that the user belongs to the system. These operations are:

- The **Boolean setAnonymity(subject, context)** method, which allows the mobile user's anonymity to be preserved, considering issues such as preventing other parties from associating the user with messages that he or she has sent or received; preventing the user's association with any communication sessions in which he or she may participate; preserving the user's privacy of location and movement information; preventing the disclosure of the relationship between a user and his or her home domain; preventing the user's association with any foreign domains that he or she may have visited; and disallowing the exposure of a user's activities by hiding his or her relationship with the visited domains. The parameters are the subject who wishes to preserve the anonymity and the context of the action or event in which we wish to establish that anonymity. If the anonymity property is established, the operation returns true.
- The **Boolean checkAnonymity(subject)** method, which controls that the mobile user's anonymity issues are preserved. This operation verifies that all data and information related to the subject are protected under anonymity. If anonymity is preserved for the subject, the operation returns true.

4.7.2. Relationships with other services

The Anonymity Service is related to the two policy services (Grid Security Policy Service and Mobile Policy Service) from which the necessary policies to establish the anonymity property in the system are obtained. This service offers an interface for the privacy service and the identity management service in order to permit the methods defined in this interface to be invoked, thus preserving the anonymity of the data and the mobile user's identity (see Fig. 8).

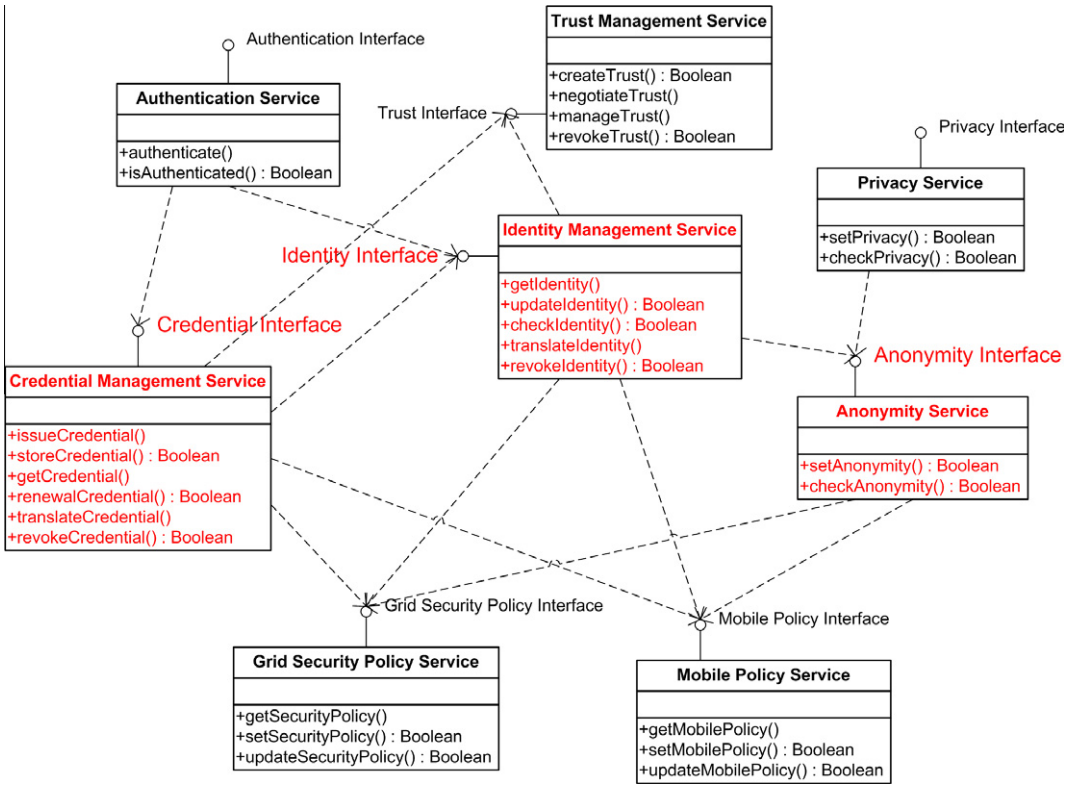


Fig. 8. Relationships between the anonymity service, Credential Management Service, identity management service and the remaining services of the architecture.

4.8. Trust management service

Trust can be generally defined as being the confidence that a party will behave in an expected manner despite the lack of ability to monitor or control that other party. Trust is normally positive, predicting a good outcome in uncertain circumstances. Trust management is the process of deciding what entities can be trusted to do what actions. In an environment in which access policy is described in terms of users' identities or required attributes, trust management consists of defining the sources of authorities for user identification, attribute assignment and possibly policy creation. In a system in which users are granted authorization tokens, the entire authorization system has been called trust management. In a system in which users can delegate some or all of their Rights to other users, the control of such delegation is part of trust management [11].

Managing trust is crucial in a dynamic Grid scenario in which Grid nodes and users join and leave the system. A mechanism with which to understand and manage the trust levels of systems and new nodes joining the Grid must therefore exist. The trust life-cycle is principally composed of three different phases: trust creation phase, trust negotiation phase, and trust management phase.

4.8.1. Trust management interface

In Grid systems, trust between entities and third parties is essential to carry out the functions and security services in the system, and this importance is enhanced by the addition of mobile devices and wireless networks. A management trust is therefore extremely important in these systems because it is responsible for establishing trust relationships between institutions, organizations and security domains that participate in the Grid. Therefore, the operations that the interface should provide are those of establishing trust between entities, negotiating a possible relationship of trust based on information and attributes of entities, another operation for the management and control of the trust relationships established, and an operation to enable the revocation and cancellation of the relations between entities. All these operations are under the control of the trust policies for each domain, entity or service involved in the trust relationship. These operations are:

- The **Boolean createTrust(parameters)** method, which establishes trust relationships between Grid entities. The creation of trust generally takes place before any trusted group is formed, and it includes mechanisms to develop trust functions and trust policies. The list of entities that form part of the trust relationship and the criteria under which such a relationship is established must be indicated in the *parameters*. If set correctly, it returns true.
- The **negotiateTrust(parameters)** method, which collects any type of information concerning entities, users, resources, domains, etc. which serves to negotiate trust relationships. Trust negotiation, on the other hand, is activated when a new system which is not trusted joins the current distributed system or group. If an entity wishes to participate in a trust relationship, it must indicate any information that it deems appropriate, to permit its addition to the relationship to be negotiated by following the trust policy established for this service. It is also necessary to indicate the entity that wishes to belong to the trust relationship of trust in *parameters*.
- The **manageTrust(trustcontext)** method, which is responsible for recalculating the trust values based on the transaction information, distribution or exchange of trust related information, updating and storing the trust information in a centralized or distributed manner. The parameter of this operation is all the information in a trust context such as might be an exchange message, stored trust data, etc.
- The **Boolean revokeTrust(entity, trustcontext)** method, which cancels the trust relations established between entities. This operation indicates the entity or entities that no longer wish to belong to the trust relationship indicated in *trustcontext*. True is returned if the revocation is successful.

4.8.2. Relationships with other services

The Trust Management Service establishes relationships with many other security services (see Fig. 7) to verify the identities of those involved in the trust relations (relation with authentication service invoking *isAuthenticated()*), to request permission to carry out the action (relation with authorization service invoking *isPermitted()*) and to delegate rights to third parties or trusted entities (relation with the delegation service invoking any of its methods). It also invokes the mobile policy interface and Grid security policy interface methods which manage and obtain security policies associated with the service. The credentials and identities that flow through the Grid must be trusted, issued, stored and managed by trusted entities, thus establishing trust relationships between domains and third parties.

4.9. Privacy service

Privacy is considered to be the ability of a mobile user to control the disclosure of personal attributes to his/her communication partners. A user's anonymity is a requisite for privacy. Privacy entails giving an entity (usually a person who is acting on his or her own behalf) the right to determine the degree to which it will interact with its environment, including the degree to which the entity is willing to share information about itself with others (anonymity).

The privacy service is primarily concerned with the policy-driven classification of personally identifiable information (PII). PII includes information such as social security number, address, age, or even soft drink preference, and this data might have different requirements for privacy protection. Service providers and service requestors may store personally identifiable information by using the privacy service. Such a service can be used to articulate and enforce a VO's privacy policy. This is generally achieved by encryption/decryption algorithms.

4.9.1. Privacy interface

If messages contain data that is of a sensitive or personal nature or that for any reason should not be visible to parties other than the sender and authorized recipients, privacy can be applied to messages, data, personal information, attributes, roles, identity, credentials, etc. which is only be visible to authorized parties. To this end, the service interface provides two operations, one to set the privacy of the entities involved in a context, and the other to verify compliance with privacy for those involved. These operations are:

- The **Boolean setPrivacy(context)** method, which establishes a private conversation between the participant entities. This signifies that only the sender and the receiver should be able to understand the messages exchanged in the conversation. If someone eavesdrops on the communication, the eavesdropper should not be able to make any sense out of it. The *context* parameter indicates the information contained in a communication, which may be requests for jobs, data that is stored in a repository, or information stored in a mobile resource. The service must follow the privacy policy associated with the service with the methods and techniques indicated for establishing privacy.
- The **Boolean checkPrivacy(context)** method, which checks whether the information exchanged between participants is carried out in a private context. The operation returns true when the context under which the communications are produced is private.

4.9.2. Relationships with other services

The privacy service uses the methods defined in the interfaces of other services to attain its goal: privacy. This service establishes relationships with many other security services (see Fig. 5) such as authentication, authorization and mobile policy and Grid Security Policy Services. The closer relationships are with the confidentiality service which helps to protect data and messages from an unauthorized entity, together with the anonymity service which permits sensitive data, identity and attributes to be protected from third parties, thus converting all this information into private information.

4.10. Credential Management Service (CMS)

Credential management is essential in a Grid context owing to the Grid's distributed and heterogeneous nature. An ideal credential-management system includes credential initiation, storage, renewal, translation, delegation, and revocation.

Credential-management systems store and manage the credentials for a variety of systems, and users can access them according to their needs. They give instructions concerning specific requirements from the credential-management systems. For typical Grid credential-management systems, mechanisms should be provided to obtain the initial credentials. This is called the initiation requirement. Similarly, secure and safe storage of credentials is equally important. In addition, the credential-management systems should be able to access and renew the credentials based on the users' demand. Other requirements which are important for Grid systems are translation, delegation, and control of the credentials.

4.10.1. Credential management interface

The CMS is a service which manages all that is related to the credentials that any entity belonging to the Grid must provide. It is therefore a service that should provide an interface with a complete set of operations that are sufficiently wide to cover the needs and capabilities for managing all credentials circulating in the Grid. These operations include issuing credentials to Grid users, the possibility of distributed storage, renewal of expired credentials, translation between different formats and types of credentials in the Grid to be understood by all services and policies, and a possible revocation of credentials. The set of operations defined in this service interface are:

- The **Credential issueCredential(parameters)** method, which should provide mechanisms to allow users to obtain the initial credentials. The Credential Management Service should provide the required credential after authenticating the user. The authentication can be based on multiple different mechanisms which may be password based, certificate based, or other types of mechanisms. The authentication service must call up this method to create an authenticated credential that is returned to the user or requesting service. To do this, the identity of the applicant, the attributes that have that entity, and the keys associated with the credential must be indicated in the *parameters*. The policy associated with this service will indicate the mechanisms and formats for generating authenticated credentials. This operation returns the credential generated.
- The **Boolean storeCredential(Credential)** method, which stores credentials to be obtained when it is necessary. As mandated by the SACRED RFC [39], the long term credentials or the private keys should be stored in the credential-management systems in a secure manner, preferably encrypted. This is an extremely important requirement, since the compromise of the long term credential would lead to disastrous consequences. This operation stores the credentials which are passed as parameter in an on-line repository which is available to any Grid service that requests it. If the credential has been stored correctly, the operation returns true.
- The **Credential getCredential(parameters)** method, which obtains credentials that have been stored in the system. This method queries the repository of credentials and searches for the credential that matches the parameters specified in *parameters*, which may be the identity, a username and password, a credential identifier, etc., depending on how the credentials repository is implemented.
- The **Boolean renewalCredential(Credential, time)** method, which should be able to handle the renewal of expired credentials, since most credentials have a specific expiry time. It is necessary to indicate the credential that we wish to renew and the new timeout value. If the renewal has been made, the operation returns true.
- The **Credential translateCredential(Credential, formatO, formatD)** method, which translates types of credentials to correct formats to allow them to be understand by the entities involved. This is important if there are multiple systems with different authentication and security mechanisms. The credentials used in one domain or realm may have to be translated into credentials in another domain which should be handled by the Credential Management Service. We indicate the credential that we wish to convert, the source format of the credential passed as an argument (formatO), and the destination format of the credential that we wish to obtain (formatD). The operation returns the credential in the destination format. It is necessary to obtain and query the policy associated with the translation of the credentials of the service.
- The **Boolean revokeCredential(Credential)** method, which should provide mechanisms to revoke credentials in the case of user compromise. The credential that we wish to cancel and remove from the repository because it has been compromised and is no longer of value in the Grid is indicated. The operation returns true if the credential has been revoked.

4.10.2. Relationships with other services

The Credential Management Service establishes relationships with some of the architecture's security services (see Fig. 8). This service invokes identity interface methods which manage the identities contained in the credentials and certificates, trust interface methods to check and manage the trust between the domain and the user owning the credentials and methods of mobile policy interface, and Grid security policy interface methods which manage and obtain security policies associated with the service. The authentication service calls up the interface methods of this service to manage the credentials used in the authentication process.

4.11. Identity management service

Identity management in computer networks is commonly described as the combination of technologies and practices for representing and recognizing entities as digital identities. In the traditional Grid computing infrastructures identity has been considered to be of a rather static nature, represented in the form of a certificate. The user and/or the organization therefore present one single certificate with the same identity to all other different organizations, who map the user's identity onto a local identity, representing the user in that specific local organization [40].

A common expression of identity in distributed systems can be seen in the research behind technologies such as the X.509 certificate deployed in traditionally static environments [41]. However, in the mobile Grid these certificates struggle to support mobile

identity in which users may need to constantly update the data in the certificate as they move into new environments. A more flexible and interoperable solution is required for the mobile Grid, and federated identity management has appeared to fill this gap. This is a new initiative whose intention is to include the necessary flexibility and interoperability to support greater user mobility within an identity solution. Federated identity allows information about users in one security domain to be provided to other organizations in a common federation (a federation is a collection of domains that have agreed trust relationships in order to correctly authenticate their respective users).

Individual identity in distributed computing applications is commonly supported through the use of PKI frameworks, and this is common in mobile environments to aid the management of quick reactions to network change [42].

4.11.1. Identity management interface

The identity of a Grid user is valid and recognized in any part of the Grid domain, principally owing to the trust relationships between domains. Identity does not vary from one domain to another, and it is possible to obtain any information from an identity whatever the domain is. In order to facilitate the management of identities in the Grid, the identity management service should provide an interface with operations to obtain the identity from the credentials submitted by users, to update the information related to an identity (roles, permissions, etc.) caused by the continuous mobility between domains, and the conversion of identities between different domains. An operation is also necessary to check the validity of identities when a particular action in the Grid is requested. Finally, it is necessary to eliminate a certain identity when an entity ceases to belong to the Grid or is no longer valid. The operations in this service which manage the identities of users and services that take part of the Mobile Grid System are the following:

- The **Identity getIdentity(Credential)** method, which obtains the identities of users or services. Given a credential, the operation obtains the identity associated with the credential stored in the repository of credentials.
- The **Boolean updateIdentity(Identity, Data)** method, which updates or modifies a specific identity. We indicate the identity that we wish to update and the data that we wish to add or alter. The information related to the identity in the Grid is updated. If the update is successful, it returns true, and false otherwise.
- The **Boolean checkIdentity(Identity)** method, which verifies whether the presented identity is valid. Given an identity, the operation checks that this identity remains valid, that it has not been compromised and that the information associated with this identity is correct. The operation returns true if all tests are appropriate.
- The **Identity translateIdentity(Identity, formatO, formatD)** method, which translates types of identities into correct formats to be understand by the entities involved. This operation converts an identity in *formatO* format into the same identity with the *formatD* format which is understood by others entities in other domains within the Grid. The operation returns the identity in destination format.
- The **Boolean revokeIdentity(Identity)** method, which revokes the identity of a user or service stored in the system. If an identity is no longer valid or no longer belongs to the Grid, it must be revoked and removed from the Grid along with all information related to that identity so that it can no longer be used within the Grid.

4.11.2. Relationships with other services

The identity management service establishes relationships with some of the architecture's security services (see Fig. 8). This service invokes anonymity interface methods which permit certain properties to be added to the identities which support anonymous identities, trust interface methods to check and manage the trust between the domain and the user owning the identities, and mobile policy interface and Grid security policy interface methods which manage and obtain security policies associated with the service. The authentication service calls up the interface methods of this service to manage the identities used in the authentication process.

4.12. Mobile Policy Service

The key issue in mobile security is that no single security solution will work given the nature of the mobile environment, and simply extending the existing security infrastructure for mobile devices is just not practical. Enterprises must treat mobile security as an independent task, and mobile-usage-specific security policies must be created and implemented as an independent task. A comprehensive risk analysis of the potential security hazards associated with the use of mobile devices should be the first step along the path of mobile device security policy creation. The creation of security policies that are specific to mobile device usage should include how to minimize the impact of a lost device: all devices should be password-protected, sensitive documents in the device should be encrypted, and automatic scripts should not be used for VPN login. Mobile device security policies should also include minimized access to limited sources by using firewalls.

The first step, therefore, is to develop reasonable security policies to govern the use of mobile devices in the network. The organization should have policies that are specific to mobile devices, and should not just attempt to apply generic security policies. It is also important that the organization educates its mobile device users with regard to security issues, including physical security. Some security policies can be enforced technologically, but others are dependent on user compliance. Mobile clients such as notebooks or PDAs should implement local protection measures such as access control, user authentication (fingerprints), antivirus protection, personal firewalls, restrictive access to hardware resources (e.g. hardrive access) and local data encryption.

4.12.1. Mobile policy interface

This service is responsible for managing the specific policies on the use of mobile devices in the Grid and should be considered in all functions and operations of security services of the architecture in which mobile devices are involved. The architecture must have a defined set of mobile policies that manage, guide and define rules and methods to be applied to any security service that requires it. To do this, we must have an interface that provides operations with which to obtain a policy associated with a service, to update information in any particular policy, or to generate and associate policies to specific services. The policies are stored in an LDAP repository that is usually implemented in the domain in which mobile devices reside. It is also possible to have a centralized repository in the Grid with a policy for each resource. This service defines a set of operations which manage mobile policies. These operations are:

- The **Policy getMobilePolicy(soa, oid, localurl)** method, which obtains the security policy or policies that govern the mobile entities. Each service or entity has an associated policy that manages and governs it, independently of the domain in which it is located. To obtain this policy it is therefore necessary to

indicate the soa and oid that uniquely defined it, and the local url in which the local policies in the LDAP are stored. If the repositories are distributed, this operation searches the mobile policy in the local repository, and if that mobile device is in a foreign domain, it will search in the domain to which it belongs. The operation returns an associated policy.

- The **Boolean setMobilePolicy(Policy, mobServices)** method, which establishes a security policy that is specific to the use of mobile devices. The operation stores a new policy (*policy*) in the LDAP repository and is associated with a mobile service or services (mobServices) to be consulted and obtained with the *getMobilePolicy* Method. If the policy has been created and stored in the repository, the operation returns true.
- The **Boolean updateMobilePolicy(Policy, Data)** method, which modifies or updates the mobile policy. This operation is responsible for updating the data stored in the mobile policy with the new data indicated as a parameter of the operation. If the update has been carried out, the operation returns true.

4.12.2. Relationships with other services

The Mobile Policy Service establishes relationships with all the security services (see Fig. 9) in the architecture, signifying that all the security services need to know and manage their own mobile policies which define the behavior in the use of mobile devices and users.

4.13. Audit service

The audit service is policy-driven and responsible for recording security–relevant events. This service is typically used by security administrators within a VO to check adherence to access control and authentication policies. Auditing requires events to be logged in a secure fashion. Logging services and secure access to logs in a distributed setting is a complex problem since logs may reside in different administrative domains. Logs should be secured and tamper-proof, and capable of ensuring message integrity. Among the events which require auditing are security events, e.g. an intrusion, which should be dealt with by the security services.

4.13.1. Audit interface

Everything that is occurring in the system must be monitored and recorded for audit and possible improvements of the system and security policies. We therefore offer a series of operations for the audit service which will record each event that has occurred with the necessary information in order to provide detailed knowledge what has happened at a particular time in the system. We will therefore have an interface with an operation of event log of the system and another operation to recover these events when they are requested by an auditor. These operations are:

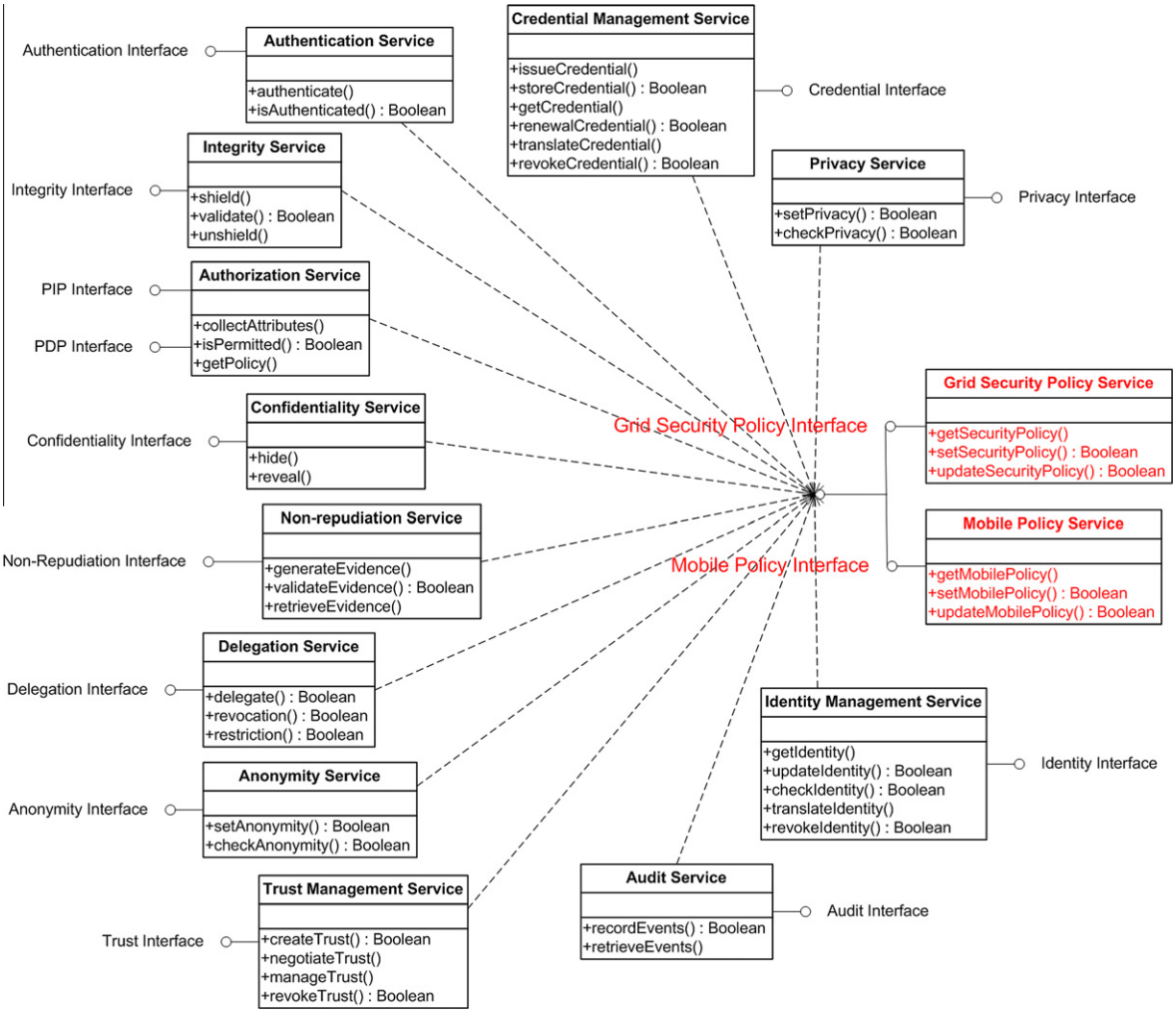


Fig. 9. Relationships between the Mobile Policy Service, Grid Security Policy service and the remaining services of the architecture.

- The **Boolean recordEvents(parameters)** method, which registries actions and events produced in the system, which are stored for their subsequent use. The action to be registered, the subject involved (applicants and recipients), the time at which the event or action occurred, and the type of security-related event must be specified in the *parameters*. The operation returns true if the register has been successfully completed. The records are created by following the audit policy that indicates how to protect the records and their availability.
- The **Event retrieveEvents(parameters)** method, which retrieves the events stored in the system. This operation recovers previously stored records in order to process and take appropriate measures in the case of finding an incident or misbehavior. It is necessary to indicate the type of event that we wish to retrieve or, depending on how the service is implemented, it can be recovered by indicating those involved in the event, or the time it was recorded. The operation returns the event requested.

4.13.2. Relationships with other services

The audit service establishes relationships with some security services (see Fig. 6). This service relates to the non-repudiation service in order to manage, record and validate parts of the events generated by the system in the form of evidences. It also relates to the Mobile Policy Service and the Grid Security Policy Service in order to obtain and manage the security policies of both the Grid and mobile resources.

4.14. Grid Security Policy Service

It is possible to define security policies in a wired network in which systems are stationary and the network structure is static. It is possible to enforce these policies or rules in such networks. However, the situation is different when we move to a mobile computing environment. In a mobile computing environment, the user will move from one device to another device or from one network to another network. These devices or networks may be similar or of different types. It is possible to define a security policy in a network with static nodes. It may also be possible to enforce such policies. However, in the case of mobile computing where nodes are roaming from one network to another, it may not be practical to define a security policy and implement it. Object security is therefore needed for mobile nodes, over and above policy-based security. In object security, objects will carry their security signatures and capabilities. This is achieved through the concept of principal. Therefore, when a device moves from network to network, the device carries the security requirement and security signature with it [43].

The Security policies include different characteristics, such as the level of protection needed for them to be applied to the various information resources that the component contains or controls, rules that determine how much trust the element places in other elements with which it communicates, cryptographic protocols that the element should use in various situations and the circumstances in which the element should apply or accept security-related patches or other updates to its own software, and so on [44].

SAML and XACML specification can be used to express access control policies, authorization assertions and authorization protocols. The XACML specification sets out the resource policy mechanism for every resource or service. SAML also has a policy mechanism, but it is very limited for the Grid, while XACML provides a more flexible mechanism that can be applied to any type of resource.

4.14.1. Grid security policy interface

This service is one of the most important because it manages the security policies that govern the Mobile Grid System, and all the security services in the architecture must have at least one associated policy to allow this service to be carried out. This service, along with the mobile service policy, completes the policy management in the security services architecture for Mobile Grid Systems. Grid services belonging to a domain or VO generally have an associated set of Grid policies, but may also have specific policies for any particular service or resource. This service provides an interface which defines simple operations to obtain a policy stored in the system, to create and associate security policies, and to update existing policies. These operations are:

- The **Policy getSecurityPolicy(soa, oid, url)** method, which obtains global security policies which govern the system. In order to indentify a uniquely policy, we must indicate the soa and oid of the policy that we wish to obtain from the LDAP repository in which the Grid security policies are stored. This operation returns an associated policy.
- The **Boolean setSecurityPolicy(Policy, services)** method, which establishes the security policy that must be fulfilled by the system. This method is used to create new security policies and integrate them within the system. These policies are associated with certain services or domains within the Grid. When the operation returns true, it indicates that the policy has been created successfully.
- The **Boolean updateSecurityPolicy(Policy, Data)** method, which modifies or updates the Grid security policy. It first discovers the location of the policy with the data defined in the policy, and then updates the policy by adding or updating new data passed in the *data* parameter. The policy is updated and made ready to be used by Grid services, returning the value true.

4.14.2. Relationships with other services

The Grid Security Policy Service establishes relationships with all security services (see Fig. 9) of the architecture, so all security services need to know and manage their own security policies which define how they must behave in a Grid environment.

5. Applicability of the reference security architecture

The reference security architecture previously defined includes a complete set of security services that ensure the fulfillment of the majority of security requirements found in the Mobile Grid environments. In this architecture we have defined security services for Mobile Grid environments that are reused in the design activity. This guarantees that the system is built in a secure environment and meets all the requirements and security needs of the system. This architecture is used as a basis from which the security architecture for the application that we are developing can be instantiated.

In the development process we have defined a task in of the design activity which carries out the integration of security into the system through a security architecture built from the security use cases identified and specified in the analysis activity. We have defined security requirement-service association rules in which the security use cases that represent security requirements can be translated into security services. These security services are integrated into the reference security architecture as instances of the abstract security services to obtain the final security architecture. The final architecture is

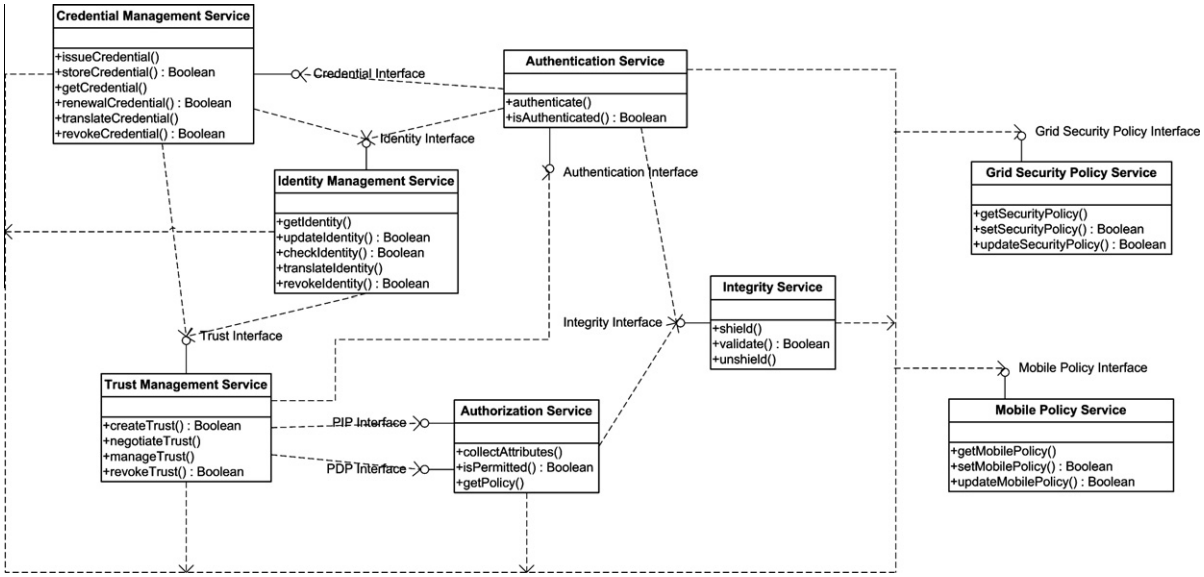


Fig. 10. Services and interfaces of security architecture instantiated for this example.

defined with all the specific security services with the help of the security requirement-service association rules from the security use cases. It is important to define the security policies of the security architecture (for each security service, virtual organization, communications, resources, mobile resources, user, etc.). This architecture must be integrated with the software architecture to obtain a security software architecture with all the security aspects required by the users and the system.

So, for example, in a first iteration of the development process we obtain security use cases such as Authentication and Authorization, which were extracted from the use cases diagram built in the analysis activity of the development process for a specific application. By applying the association rules defined between security requirements and security services, we obtain that the security services that we must consider for the security architecture of this application, which were obtained from the security use cases identified in the previous activity for this first iteration, are: Authorization, Authentication, Credential Management, Identity Management, Trust Management and Integrity, together with the Grid Security Policy service and the Mobile Policy Service which are needed to manage the different policies of the system. The resulting architecture for instantiating the reference security architecture for this application in this first iteration is shown in Fig. 10, in which we can see the class diagram of the services and their interfaces, together with the relationships between them.

Moreover, in the design activity we have defined V&V aspects in a task which validates whether the designed software architecture covers, fulfils and considers all the requirements (contained in the analysis model artifact) specified in the analysis activity. This V&V task also validates that the security and attack scenarios considered in the current iteration are solved, and identifies potential conflicts of the solution (designed architecture) with regard to functional or quality requirements. It also verifies the traceability of artefacts, studying whether the analysis artefacts have been considered and correctly generated in the design artefacts. What is more, it verifies that the other scenarios are fulfilled with the architecture design, identifying possible conflicts, studying their scope and analyzing the opportunities to correct and deal with them.

This security architecture will be built in the construction activity, implementing the services and interfaces, and generating code for the interface methods, together with its parameters to establish communication and message exchange between the different services inside and outside the architecture.

6. Conclusions

This paper presents our security architecture with which to handle various security problems in the Grid. How to construct a secure Grid system is also an important task owing to the high complexity of the Grid computing system. It is difficult to incorporate mobile devices in the Grid in a secure manner so that the impact is minimal and transparent to the user. It is therefore necessary to develop and define a process for developing a system based on Grid and mobile technology, considering the security peculiarities and needs of this kind of system from the early stages of development. The Grid needs to build a unified security system that supports all security functions, and a service-oriented security architecture that defines a wide set of security services and covers all the security requirements for the mobile Grid environments is therefore fundamental for the construction of a security model for this kind of systems.

An important stage in the development process is the design activity, which focuses on ensuring that the system's security and functional requirements are fulfilled, covered and validated with the design, on the one hand, of a security architecture, and the other hand, of a software architecture. In our approach, this activity is supported by a reference security architecture which covers and fulfils the security requirements of the Mobile Grid System specified in the analysis activity. This reference security architecture is instantiated in a concrete security architecture (depending on the security requirements needed) and is integrated with the software architecture designed, thus obtaining a Secure Software Architecture for Mobile Grid Systems which is an input artefact for the construction activity. It is important to verify and validate the process applied in this activity and the results obtained to assure the quality of the final system.

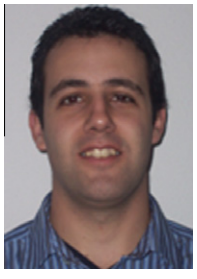
As future work, we shall complete the formal development process in order to describe the phases, activities and tasks with the SPEM specification. We shall carry out an in–depth study of the proposed security services, identifying security technologies, protocols and mechanisms that can be used in each service. We are applying this development process to a real case in which we will build a specific security architecture for this application from the reference architecture. The resulting security architecture will be implemented in the construction activity, and the interfaces, services and methods will be implemented with a specific programming language and tools and technologies for Grid systems (Globus, PERMIS, etc.). An adaptation of this proposal to Cloud computing, a technology which provides computer services through the Internet, could also be studied.

Acknowledgments

This research is part of the following projects: QUASIMODO (PAC08-0157-0668), SISTEMAS (PII2I09-0150-3135) and SEGMENT (HITO-09-138) financed by the "Viceconsejería de Ciencia y Tecnología de la Junta de Comunidades de Castilla-La Mancha" (Spain) and FEDER, and MEDUSAS (IDI-20090557) and BUSINESS (PET2008-0136) financed by the "Ministerio de Ciencia e Innovación (CDTI)" (Spain).

References

[1] I. Foster et al., The anatomy of the grid: enabling scalable virtual organizations, in: Proceedings of the 7th International Euro-Par Conference Manchester on Parallel Processing, Springer-Verlag, 2001, pp. 1–4.
[2] D.E. Millard, et al., Experiences with writing grid clients for mobile devices, in: Proceedings of the 1st International ELeGI Conference on Advanced Technology for Enhanced Learning BCS Electronic Workshops in Computing (eWiC), 2005.
[3] W.A. Lin, Building a unified grid, Part 1: Grid architecture in the Telescience Project, National Center for Microscopy and Imaging Research, 2005.
[4] L. Srinivasan, J. Treadwell, An Overview of Service-oriented Architecture, Web Services and Grid Computing, HP Software Global Business Unit, 2005.
[5] L. Bass et al., Security and survivability reasoning frameworks and architectural design tactics, SEI, 2004.
[6] J. Jürjens, Secure Systems Development with UML, Springer-Verlag, 2004.
[7] T. Lodderstedt et al., SecureUML: A UML-Based Modeling Language for Model-Driven Security, Springer, 2002. pp. 426–441.
[8] R. Breu et al., Key issues of a formally based process model for security engineering, in: Proceedings of International Conference on Software and Systems Engineering and their Applications, 2003.
[9] C.B. Haley et al., A framework for security requirements engineering, in: Proceedings of Software Engineering for Secure Systems Workshop, 2006, pp. 35–42.
[10] H. Mouratidis, P. Giorgini, Integrating Security and Software Engineering: Advances and Future Vision, IGI Global, 2006.
[11] M. Humphrey et al., Security for Grids, Lawrence Berkeley National Laboratory, Paper LBNL-54853, 2005.
[12] A. Chakrabarti et al., Grid computing security: a taxonomy, IEEE Security and Privacy 6 (1) (2008) 44–51.
[13] R. Kolonay, M. Sobolewski, Grid interactive service-oriented programming environment, in: Proceedings of Concurrent Engineering: The Worldwide Engineering Grid, Press and Springer-Verlag, 2004, pp. 97–102.
[14] H. Dail et al., Scheduling in the Grid Application Development Software Project, Grid resource management:state of the art and future trends, 2004, pp. 73–98 (Chapter 1).
[15] D.G. Rosado et al., PSecGCM: Process for the development of Secure Grid Computing based Systems with Mobile devices, in: Proceedings of International Conference on Availability, Reliability and Security (ARES 2008), IEEE Computer Society, 2008, pp. 136–143.
[16] IEEE, IEEE Standard for Software Verification and Validation, IEEE Std 1012™-2004, 2004.
[17] D.G. Rosado et al., Engineering Process Based On Grid Use Cases For Mobile Grid Systems, in: Proceedings of the 3rd International Conference on Software and Data Technologies, ICSOFT 2008, 2008, pp. 146–151.
[18] D.G. Rosado et al., Obtaining security requirements for a mobile grid system, International Journal of Grid and High Performance Computing 1 (3) (2009) 1–17.
[19] D.G. Rosado et al., Reusable security use cases for mobile grid environments, in: Proceedings of the Workshop on Software Engineering for Secure Systems, Conjunction with the 31st International Conference on Software Engineering, 2009, pp. 1–8.
[20] D.G. Rosado et al., Towards an UML Extension of Reusable Secure Use Cases for Mobile Grid Systems, IEICE Transactions on Information and Systems, 2009 (under review).
[21] D.G. Rosado et al., Applying a UML extension to build use cases diagrams in a secure mobile Grid application, in: Proceedings of the 5th International Workshop on Foundations and Practices of UML, Conjunction with the 28th International Conference on Conceptual Modelling, ER 2009, LNCS, vol. 5833, 2009, pp. 126–136.
[22] D.G. Rosado et al., Analysis of secure mobile grid systems: a systematic approach, Information and Software Technology 52 (2010) 517–536.
[23] T. Phan et al., Challenge: integrating mobile wireless devices into the computational grid, in: Proceedings of the 8th Annual International Conference on Mobile Computing and Networking (MobiCom'02), ACM Press, 2002, pp. 271–278.
[24] B.A.M.H. Clarke, Beyond the 'Device as Portal': meeting the requirements of wireless and mobile devices in the legion grid computing system, in: Proceedings of the Parallel and Distributed Computing Issues in Wireless Networks and Mobile Computing at the International Parallel and Distributed Processing Symposium, IEEE Press, 2002.
[25] Open Grid Forum, The Open Grid Services Architecture, Version 1.5, 2006.
[26] Globus Project, Grid Security Infrastructure (GSI), 2005. <www.globus.org/security>.
[27] EGEE Middleware Design Team, EGEE Middleware Architecture, 2004. <https://edms.cern.ch/document/476451/>.
[28] Enterprise Grid Alliance Security Working Group, Enterprise Grid Security Requirements Version 1.0, 2005.
[29] S. Chapin et al., A new model of security for metasystems, Future Generation Computer Systems 15 (5-6) (1999) 713–722.
[30] M. van Steen et al., Globe: A Wide-Area Distributed System, IEEE Concurrency, 1999.
[31] E. Belani et al., CRISIS: a wide area security architecture, in: Proceedings of the Seventh USENIX Security Symposium, 1998.
[32] M. Colombo et al., Fine grained access control with trust and reputation management for globus, in: Proceedings of the OTM Confederated International Conferences, LNCS, 2007, pp. 1505–1515.
[33] D. Fais et al., An implementation of role-base trust management extended with weights on mobile devices, Electronic Notes in Theoretical Computer Science 244 (2009) 53–65.
[34] OMG, Software and Systems Process Engineering Meta-Model Specification (SPEM) 2.0, 2008.
[35] W.A. Jansen, Authenticating users on handheld devices, in: Proceedings of the Canadian Information Technology Security Symposium, 2003.
[36] Y. Xiao, Security in Distributed, Grid, Mobile, and Pervasive Computing, Auerbach Publications, 2007.
[37] D. Samfat et al., Untreacibility in mobile networks, in: Proceedings of the ACM International Conference on Mobile Computing and Networking, 1995.
[38] H. Imai et al. (Eds.), Wireless Communications Security, Artech House Publishers, 2005.
[39] A. Arsenault et al., Securely Available Credentials – Requirements, IETF RFC 3157, 2001.
[40] H. Mikkonen, M. Silander, Federated identity management for grids, in: Proceedings of the International conference on Networking and Services (ICNS'06), 2006, p. 69.
[41] ITU, ITU_T Recommendation X.509. Information Technology – Open Systems Interconnection – The Directory: Public-key and Attribute Certificate Frameworks, 2005.
[42] K. Bayarou et al., Towards certificate-based authentication for future mobile communications, Wireless Personal Communications 29 (3-4) (2004) 283–301.
[43] A. Talukder, R. Yavagal, Security issues in mobile computing, Mobile Computing, McGraw-Hill Professional, 2006. Chapter 18.
[44] A. Sajjad et al., AutoMAGI – an Autonomic middleware for enabling Mobile Access to Grid Infrastructure, in: Proceedings of the Joint International Conference on Autonomic and Autonomous Systems and International Conference on Networking and Services – (icas-icns'05), 2005, pp. 73–73.



David G. Rosado (David.GRosado@uclm.es) holds a Ph.D. in Computer Science from University of Castilla-La Mancha and has an MSc in Computer Science from the University of Málaga (Spain). His research activities are focused on security architectures for Information Systems and Mobile Grid Computing. He has published several papers in national and international conferences on these subjects. He is a member of the GSyA research group of the Information Systems and Technologies Department at the University of Castilla-La Mancha, in Ciudad Real, Spain.

Eduardo Fernández-Medina (Eduardo.fdezmedina@uclm.es) holds a PhD. and an MSc. in Computer Science from the University of Sevilla. He is associate Professor at the Escuela Superior de Informática of the University of Castilla-La Mancha at Ciudad Real (Spain), his research activity being in the field of security in databases, datawarehouses, web services and information systems, and also in security metrics. Fernández-Medina is co-editor of several books and chapter books on these subjects, and has several dozens of papers in national and international conferences (DEXA, CAISE, UML, ER, etc.). Author of several manuscripts in national and international journals (Information Software Technology, Computers And Security, Information Systems Security, etc.), he is a member of the GSyA research group of the Information Systems and Technologies Department at the University of Castilla-La Mancha, in Ciudad Real, Spain. He belongs to various professional and research associations (ATI, AEC, ISO, IFIP WG11.3 etc.).

Javier Lopez (jlm@lcc.uma.es) received his M.S. and Ph.D. degrees in computer science in 1992 and 2000, respectively, from the University of Malaga, where he currently is a full professor. His research activities are mainly focused on network security and critical information infrastructures, and he leads national and international research projects in those areas. He is also Co-Editor in Chief of Springer's International Journal of Information Security (IJIS), a member of the editorial boards of international journals, and the Spanish representative on the IFIP Technical Committee 11 on security and protection in information systems.