Propagating Trust and Privacy Aspects in Federated Identity Management Scenarios

Latifa Boursas[1] and Helmut Reiser[2]

[1] Munich Network Management Team
Technische Universität München
boursas@tum.de,
http://www.nm.ifi.lmu.de/~boursas/
[2] Leibniz Supercomputing Center
Boltzmannstr. 1
D-85748 Garching Germany
reiser@lrz.de,
http://www.nm.ifi.lmu.de/~reiser/

Abstract. This paper presents a trust-management application for trust based relationships in federated identity management (FIM). It empowers discovery, technical interpretation of the meaning of trust, dynamic calculation of trust levels and afterwards static composition of trustworthy relationships between Identity and Service providers, which can be accomplished through the use of service-specific XACML policies to enforce the required trust and privacy aspects.

1 The Problem Statement

Nowadays most of the emerging approaches to digital identity management have been developed to address the distributed user data management problem, by providing the feature of the federation process while each partner service has its own privacy constraints and security policies. In addition to deciding what right each user has to access a Service Provider (SP), it is also possible to the user to decide if he wants to access this service provider without re-authentication. The condition to enable this right is that the user must authenticate at an Identity Provider (IDP) who is recognised and accepted by the SP. Therefore, providing efficient and seamless interrelationship between IDPs and SPs in inter-organization scenarios, based on FIM, requires building trust-based relationships that enable IDPs to securely distribute the user's identity information, and the SP to manage and control the access set to the services it provides. Consequently, those trust relationships, which allow identity and policy information to flow between the participating SPs, need to be formed quickly and efficiently to maximize productivity and eliminate the manual processes that often take place nowadays.

In today's federations, trust between IDPs and SPs is typically established on a contractual basis and thus quite static by building the so-called *trust communities*. The other technical aspects of trust are limited to authenticating and authorizing users, but secure FIM systems should not only authenticate users, but also allow them to delegate their rights and beliefs to other users securely. However, we want to achieve more dynamic

trust relationships and especially involve the aspects of reputation management which allow the peer to reflect its own experience as well as the experience of other known peers with the new peer; and thus building dynamic trust relationships based on recommendations of other peers, as considering the human factors beside the technical aspects is more and more becoming a crucial question.

The objective of this work is to build such trust relationships between group of peers, within organizations, who meet the trust requirements through individual experience and sharing experiences with other peers with similar preferences, as most of the answers to such issues often depend on whether the peer we want to communicate with, is someone inside or outside the organization and also depends on how a set of organizations can be defined in connection with the IDP and the SP in FIM. This issue relate, more precisely, to designing an appropriate algorithm for dynamically computing and interpreting the value representing the *trust level* between the peers. Therefore, the algorithm will be based on a collection of attributes that are basically created, along with an identity of the peer, during a process of the interaction, as they tend to identify the other party as well as the level of trust in order to estimate carefully how it is likely to behave in a given situation depending on the rewards for being trustworthy. In the second stage of our solution, the outputs of the algorithm can be afterwards useful in the policy-based privacy rules for constructing an efficient threshold decision point, as it is the case in many FIM-Scenarios that the users' data can be only released if a certain threshold of the trust level is exceeded.

The rest of the paper is organized as follows: Section 2 introduces a detailed analysis to the requirements on the IDP as well as on the SP side that may occur in our trust model. In section 3 we explain our solution based on the recursive trust algorithm, including the acquaintance graph and the different related mechanisms and aspects that may be integrated within the proposed algorithm to provide the desired trust model. Section 4 provides an analysis about the LDAP schema and the DIT definition that will help define the trust relationships among the principals. Implementation analysis as well as evidence will conclude this section. Section 5 provides an overview of the state of the art and similar approaches for estimating the value of trust. Finally Section 6 concludes the current status and discusses future research directions.

2 Requirement analysis of trust in Federated Identity Management

Usually the main kinds of peers - known as principals - connected by those trust relationships are based on their particular identity lifetimes: people who can make assertions affiliated to their long-lived IDs; computers can make close evidence of correct operation from their IP-Addresses; and organizations are generally a set of people and computers and can benefit from this collection to build trust relationship. The Credentials describe each kind of principal, often as attributes attached to that principal and its relationships to other principals [8].

The focus of this solution is to figure out the way appropriate correspondent can be selected, when calculating the trust level according to the user's respectively the Service Provider's trust requirements. This approach for modelling trust is realised as an algorithm in defining, representing, calculating, and propagating the value of trust from which we can quantitatively read the relationship of any two entities in our Framework. While collecting all the necessary requirements for the trust calculation algorithm, it is primordial to investigate the following issues on the IDP side as well as on the SP side:

- The user identity mapping which allows SPs to correlate personal information about the user in a way that otherwise would not be possible. When the IDP consents to identity mapping, the SP must accept a given policy for how mapping can be used to correlate data linked to the different identities in the corresponding IDP. Consequently, the SP's failure to adhere to the privacy policy for identity mapping can cause distrust.
- Validation of SP's term of use, privacy policies, shipping conditions, as stated and agreed in the formal contract and service level agreements.
- Previous experience with the service and with other services within the same SP.
- Trust and confidence of other users (for example within the same IDP) in that SP.

2.1 Requirements of the Service Provider on the users and their Identity Provider

On the one hand service provider provides services for end users, and on the other hand they act as a relying party to validate credentials issued by a trusted identity partner, on the basis of which they provide services to that trusted identity. Trust can be established for users and Identity Providers through assurances that they follow the recommended practices of the SP for handling authentication credentials with adequate care, and by having a history with the service provider that is free of security incidents. The SP, consequently, may require investigations, for instance on:

- The way credentials are issued to the users and whether the IDP has implemented adequate procedures for registering users and for issuing their credentials in a trustful way.
- Experience of the SP with the user, for instance experience on having a history without registration error because of unpaid bill or delivery rejections.
- Experience of the SP with the users' IDP.
- Rating and opinions of other known Service Providers about the concerned IDP.

3 Solution

This contribution presents a trust model for dynamic calculation of the level trust to convey that trust in FIM has in some specific manner a relative aspect, like A may trust B with respect to X, and to beforehand not necessary known principals and afterwards use statically these trust values for future interactions with other new principals. In the

first stage of our model, we represent the principals as nodes into a trust graph, where the weight of the connections between the nodes is estimated according to an algorithm that is based on a collection of attributes. These attribute are created, along with the identity of the principal, during a process of the interaction, as they tend to identify the other party as well as the level of trust in order to estimate carefully how it is likely to behave in a given situation [4]. Additionally we extend the algorithm and apply the rules recursively on all possible situations that may happen in the context of identity and service federation. The outputs of the algorithm can be used, in the second stage of this trust model, in the policy-based privacy rules for constructing an efficient threshold decision point, and accordingly releasing the users' data if a certain threshold of the trust level is exceeded.

For presenting and linking the principals in trustworthy groups, we use a dedicated schema in order to allow them to create and interlink statements about whom they know and how they put trust in new principals. Within this schema, new attributes are described to integrate and extend the predefined privacy and trust aspects. Similarly to the ontology based trust model from Golbeck [13], the levels of trust roughly go from 1, corresponding to an absolute distrust, to 9 corresponding to an absolute trust. This estimated level of trust can be specified in general, or limited to a specific use or purpose, which are called trust situations, in order to refine the communication between principals. For instance a SP offers a set of services which can be collected into situations; this leads to hierarchies and chains of situations that will be in so doing associated to trust values.

In addition to the schema-extension with trust attributes, our proposed trust model uses an algorithm for calculating the trust level and adds threshold properties which can be included as rules within the service-specific *eXtensible Access Control Markup Language* (XACML) policies (the Attribute Release Policies *ARP*) that are basically used for controlling the flow of user's personal information [6] (see Section 4).

3.1 Representation of the trust acquaintance graph

As mentioned above, using these rules by means of the defined principal's trust profile, a simple recursive algorithm, can be applied to calculate the average value of trust and propagate it for future iterations. The trust algorithm is based on a graph of the principal's acquaintances, which can be defined as a set of nodes representing the principals and a set of edges that connect them and define the trust relationships between them. We consider that pairs of apparently distant principals are actually connected by a chain of intermediate acquaintances.

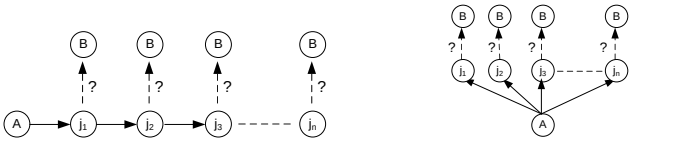
The investigated path between the pairs of principals is, consequently, an alternating sequence of nodes and edges, beginning at a node A and ending at a node B, and which does not visit any node more than once.

Linear representation of the acquaintance graph PGP Web of Trust [18] can be viewed as a directed graph where the nodes are the keys, and the edges are the signa-

tures. The path from A to B can be found by tracing the set of the intermediate signed keys, such that for every keyring, it is very important that a node signs the key of other in order to be able to find such paths. Similarly, in our linear representation of the graph, as illustrated in figure 1.a, we first seek to enumerate all nodes that are known to node A, we follow a sequence of edges to *walk* through the graph, composed of pairs of nodes denoted (A, j_i), until we reach a node which has a directed edge to node B. The path from node A to node B is then a sequence of edges (A, j_1), (A, j_2), ... (A, j_n), and its weight corresponds to the value of trust that has to be measured in function of the weight of each intermediate edge.

This graph, which is known as digraph [1], is pretty simple as it has no loops and at most one edge between any pair of nodes, where we distinguish *out-degree* edges, the edges leaving a node, and *in-degree* edges, the edges entering a node. The presentation of this graph aims at keeping the search complexity of the algorithm linear, and the pointer does not have necessarily to go through all the nodes on the path; however, one can expect some intermediate nodes to contain links to the unknown node, and as a soon as such a link has been found, the search path comes to an ending point. But from the performance perspective, this representation suffers from a fatal problem: The assumption for accepting a node B as trustworthy is estimated just for a given path and does not allow overall analysis for highly efficient distributed communities, because in most cases all participants need only check a small, local subset of the global trust graph.

Out-Degree representation of the graph In this model of the graph, we illustrate our modified representation of the graph and show that significant improvements can be obtained using an exclusively oriented graph (a directed acyclic graph [10]) by means of the edges leaving the node A (out-degree) until reaching the node B. As can be seen in figure 1.b, the distance from node A to node B is only based of the out-degree of the neighbours of node A that have to be visited as this graph representation is intended to serve as an overall assessment over all the existing nodes. The path search works as follow: The pointer selects the first unvisited node, pick up the weight of its directed edge, and move on in order to find the next unvisited node, always from the same starting node. These steps will be repeated until there will be no more unvisited nodes on the graph. The trust algorithm presented in the following section relies on this structure of the graph.



a. Linear representation b. Out-Degree representation

Fig. 1. Representation of the trust acquaintance graph

3.2 The trust algorithm

We now describe in details how the algorithm, illustrated in figure 2, works to compute the weight of requested edge from node A to node B:

- Check from past experiences if the principal A with n neighbours already made a direct transaction with the new principal B. If so, we ignore other weights and use the weight of this relationship as its value of trust. Because if a line connects two nodes, they are considered to be *adjacent*, as the graph-theoretic distance between two nodes is defined as the length of the shortest path between them [1].
- If A has not directly interacted with B, the value of trust is determined by a weighted average of the values for each of its neighbours with past experience with B. The calculated trust $t_{AB}(s)$ from principal A to principal B in a situation s (s represents a trust situation within a set of S situations) is given by the following function:

$$t_{AB}(s) = \frac{\sum_{j=0}^{n} \begin{pmatrix} t_{jB} \cdot t_{Aj} \text{ if } t_{Aj} \geq t_{jB} \\ (t_{Aj}^2) \quad \text{if } t_{Aj} < t_{jB} \end{pmatrix}}{\sum_{j=0}^{n} t_{Aj}}$$

  Where A has n neighbours with past experience with B. As pointed out earlier, by calculating the average, in this formula, we ensure that we do not trust someone down the line more than we trust any intermediary.
  Note that for the case where all the neighbouring edges have a highest weight t_{max} (the highest level of trust in our schema) to node B, without comparing the weight of the edges, t_{AB} will be:

$$t_{AB} = \frac{\sum_{j=0}^{n} t_{max}^2}{\sum_{j=0}^{n} t_{max}} = \frac{n \cdot t_{max}2}{n \cdot t_{max}} = t_{max}$$

  In the same manner for the opposite case where all the neighbouring nodes have the lowest weight t_{min} to node B, following the comparison stated above, the value of t_{AB} is:

$$t_{AB} = \frac{\sum_{j=0}^{n} t_{Aj}^2}{\sum_{j=0}^{n} t_{Aj}} = \frac{t_{A1}^2 + t_{A2}^2 + ... + t_{An}^2}{t_{A1} + t_{A2} + ... + t_{An}} = Cte \text{ where } t_{min} < Cte < t_{max}$$

  Consequently, this formula states that, for a finite subset of n nodes the sums of the trust values may be computed incrementally and it follows that the final trust level between two nodes can be computed using $O(n)$.
- Up to now, we just put the meaning of trust that one entity has in another in some quantitative real-values, in order to process a restricted objective. We still have to generalise the trust level, which we want to associate to the principal. Therefore we have to take into account all the possible kinds of communications and requests that might occur. We explored how those trust-level oriented applications may be chained together, and we thought about capturing a simple average of the trust weight around the sum of those thinkable subject areas:

Let s be a positive integer and let S be a set of the situations in which a request for calculating trust might be sent. $t_{AB}(s)$ represents the trust A has in B in a situation s ($s \in |S|$):

$$t_{AB} = \lfloor \frac{\sum_{s=0}^{S} t_{AB}(s)}{|S|} \rfloor$$

- Update the new trust relationships that this propagation can allow us to carry that conclusion one step further for future requests.

On the basis of the concrete mapping of these computed values to the trust definition in the above-named schema, a decision whether the user A accepts or rejects the communication with the user B can be then taken. Direct relationships between individual objects in the identity Repository will then contain an explicitly expressed trust value.

Beyond knowing that a given user explicitly trusts another, this trust data basis can be used afterwards to infer the trust that one user should have for individuals to whom they are not directly connected.

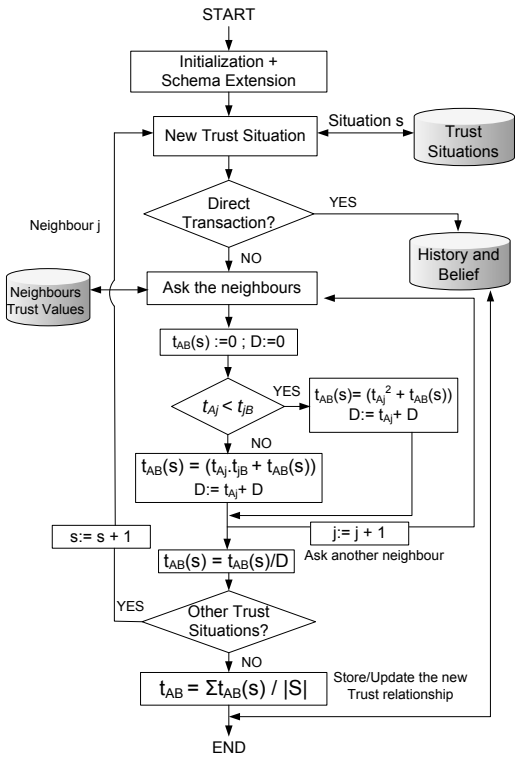


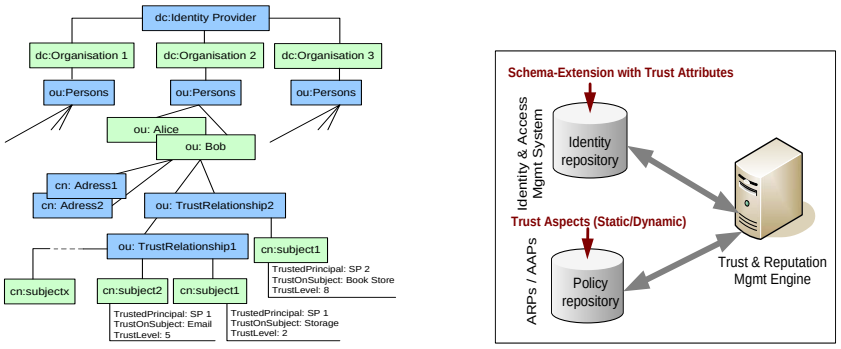
Fig. 2. The recursive trust algorithm

4 Evidence the Solution Works

In this section we provide evidence about the feasibility of the recommended trust model, which reflects the common need for organizations to outsource their users' identity data, to third parties but provides restricted access according to the trust requirements. We show how the resulting trust values from the algorithm can be integrated within the Attribute Release Policies ARP on the IDP side, as well as the design of an appropriate schema including new trust attributes and the processing of the prototype implementation.

4.1 Schema Definition

Our trust schema that has been designed in the identity repository, based on Novell's eDirectory LDAP Server [12], adds properties within a domain of *Person* as well as defining completely a new help class for expressing the value of trust among principals. The new help class *TrustRelationship* has mainly three attributes: *trustedPrincipal* indicates the principal being trusted, *trustOnSubject* indicates the subject that the trust is about, and *TrustLevel* indicates the level of Trust for this principal on the specified subject area. Figure 3.a illustrates the internal structure of the LDAP server; the so-called Directory Information Tree (DIT) that has been defined to represent the principals within the Identity Provider.



a. The Directory Information Tree b. Trust level calculation processing

Fig. 3. The integration of the trust level values

As can be observed, unlike many traditional directory conventions which aim at storing user objects as leaf objects, the user objects are stored into container objects so that further objects representing the trust relationships of the user to the corresponding SP, can be stored beneath them, usually beside other sub-objects representing for instance the user addresses. This design shows a lot of flexibility regarding storing multiple

objects of the same type without having to extend the schema and define new attributes for each new created object separately.

We had several options for creating a repository for storing the trust relationships. By using multiple trust attributes in the user object (TrustLevel1, TrustLevel2, ...) mapped on (trustOnSubject1, trustOnSubject2, ...), we encountered the problem that only a limited number of trust relationships can be stored, and the schema had to be constantly extended with additional attributes especially when it has to do with a new IDP or SP for example offering new services that have to be incorporated into new trust situations. This solution also made linking the values of trust to the appropriate SP and to the services less flexible, because setting them all at the same level affects everything underneath it, and thus, it became harder to determine those trust relationships.

The optimal option was to create a subtree or branch point by placing a container of the trust relationship under the person container. For example, we create a subtree for Service Provider 1 under the suffix: ($dc = identityProvider, dc = organization1, ou = persons, ou = Bob, cn = TrustRelationship1$). Thus, all the instances of the trust relationships for a designated subject can be placed in an unrestricted way under the container *TrustRelationship1* (subject1, subject2, ...). The single-valued attributes *TrustLevel* can than serve to specify the calculated value of trust related to the Subject stored in *trustOnSubject*. A second advantage of this method is that every trust relationship is set separately from the rest, and the trust level can be easily filtered out my means of the name of the SP as well as the Subject area.

4.2 Implementation

The actual trust level is calculated dynamically at runtime from the various static and dynamic trust components that are stored within the trust relationship sub-objects, as illustrated in the DIT in section 4.1. Decisions made on such calculation results and their varying input values will be logged for auditing purposes as well as for future iterations of the algorithm.

As can be seen in figure 3.b, our workflows involve three main steps on deciding upon the request made, and permit the users data release if the requester has been accepted as trustworthy partner:

1. First, we specify a search mechanism in the form of a search filter based on the trust attributes, in addition to the name and the Identifier of the Service Provider who attempts to access the user's data by his Identity Provider. The search filter strives to collect machine readable encoded information about past interactions with the new Service Provider and make it available to the trust algorithm.
2. Secondly, the recursive trust algorithm is applied to make estimation about the level of trust. The algorithm performs incremental alignment in real time of two series, as one is performed on the set of the neighbours' recommendations and the second is performed on the set of the trust situations.
3. Finally, the computed trust level value will serve as a parameter for the XACML-based ARPs [6] reception thresholds in the policy repository in order to assess the

limits imposed by the IDP for releasing the user's data, for example, if the trust level for the requesting service is ≥ 6.

In our implementation, we are using Novell eDirectory LDAP Server [12] for simulating the Identity Repository-based Trust Schema of the Identity Provider as described in section 4.1. We implemented the trust recursive algorithm in the *Trust & Reputation Mgmt Engine* running on perl, in order to take advantage of the perl-ldap distribution allow complex and cross-platform searches of LDAP directories.

5 Competitive Approaches

The traditional approach for solving the problem of trust in FIM has been Single Sign On (SSO) [2], the centralization of access control information into one server for enabling the users to authenticate once and gain access to the resources of multiple software systems. SSO gives in this way the possibility to build large and expandable circles of trust where many different service providers can be deployed. Consequently every application needs to be SSO enabled by programming to the proprietary API and by using the same protocols, in order to enable an application to assert the identity of a user to another, thereby avoiding the need for redundant authentication. But this constraint is rather impractical for FIM-Systems because partners may not agree on SSO software, and it is not often possible to have a unified mechanism among providers, because authentication technologies and access policies are often fragmented and dissimilar.

Formal logical models that have been used in the context of cryptography and authentication, such as *Pretty Good Privacy* (PGP) [18] and X.509 [16] solutions, attempt to solve of the trust management issues and introduce the notion of trust level (Web of Trust). But it is notable that in both solutions the unique notion of trust covers only the aspects of the authentication and the verification of the user's credentials.

There are many additional commercial solutions in FIM environment and web services, such as Liberty Alliance [9] that has defined the notion of *circle of trust* for rating how much one entity trusts another. This approach has put deep emphasis on privacy aspects inside one organization, but for our needs for establishing inter-organization trust, decentralized or federated storage of trust information related to the identity of the principals are relatively missing. Blaze et al. [7] have introduced a trust-management system as a standard interface for an application to ask if a particular requested action should be allowed, and a standard language for writing distributed and inter-operable policies and credentials, but it remains to investigate whether this interface can be applicable as a pre-processor in the field of federated identity management.

The trust formulation process such as in Web Based Social Networks [15] is generally computed via a selection of default calculations. Each trust concept has associated with it a specific trust calculation set, i.e. reliability is calculated from the reliability calculation set. However, the interpretation and the integration of the substantial technical concepts for expressing and estimating trust (Competence and honesty [5], Reliability [4],

expressing Reputation and History [17], Faith [14] and Belief [11]) remain inside the scope of our actual research.

6 Current Status and Next Steps

This contribution provides a basis for assessing the realisation of putting dynamically the meaning of trust in some real value, by means of the recursive trust algorithm, which has been designed to be flexibly extended to new trust situations. We exploited in particular the enhanced functionality and the relevance of the trust level to enforce a fine-grained access control and organisations' privacy or data protection in FIM environment from the IDP side. But similarly, the SP will also need to trust the IDP, and most importantly the end users whose identities are often subject of federation.

We plan to extend the IDP-based version of our prototype, such that the SP will also have the possibility to rate the user's behaviour and to store its experience with the user in its customer records. This data can be adequately used to maintain and refine the Attribute Acceptance Policies AAPs, which play similar roles as the ARPs but on the service provider side, by checking the validity of the information provided by the IDP for the provisioning of the requested service [3].

Further goals of our work are to predict an unknown trust or distrust value between any two principals that have no previous experience between each other and no one of their neighbours did, using the entries available in the so-defined trust matrix M. Hence, a complementary research is to produce this final matrix from which we can read off dynamically the computed trust value of any two principals in the mentioned FIM-application.

References

1. F. Maghoul A. Broder, R. Kumar. Graph structure in the web. In *Proceedings of the 9th international World Wide Web conference: The international journal of computer and telecommunications networking*, pages 309 – 320, Amsterdam, The Netherlands, February 2000.
2. C.J. Mitchell A. Pashalidis. *A Taxonomy of Single Sign-On Systems*. Springer Berlin / Heidelberg, 2003.
3. L. Boursas and W. Hommel. Policy-based Service Provisioning and Dynamic Trust Management in Identity Federations. In *Proceedings of IEEE International Conference on Communications (ICC 2006)*, Istanbul, Turkey, Juni 2006.
4. J. Golbeck, J. Hendler, and B. Parsia. Trust Networks on the Semantic Web. In *12th International Web Conference (WWW03)*, May 2003.
5. T. Grandison and Sloman. Using A Survey of Trust in Internet Applications. In *IEEE Communications Surveys*, pages 2–16, 2000.
6. W. Hommel. Using XACML for Privacy Control in SAML-based Identity Federations. In *Proceedings of the 9th Conference on Communications and Multimedia Security (CMS 2005)*, Salzburg, Austria, September 2005.
7. M. Blaze J. Lacy and J. Feigenbaum. Decentralised Trust Management. In *The IEEE Symposium on Security and Privacy*, 1996.

8. R. Kahre and A. Rifkin. Trust Management on the world wide web. In *First Monday - Peer-reviewed Journal on the Internet*, 1998.
9. Liberty Alliance Project. http://www.projectliberty.org/.
10. B. D. McKay. Acyclic Digraphs and Eigenvalues of (0; 1)-Matrices. In *Journal of Integer Sequences, Vol. 7*, 2004.
11. H.D. McKnight and N.L Chervany. The Meanings of Trust. Technical Report 94-04, Department Carlson School of Management, University of Minnesota, 1996.
12. Novell eDirectory and Identity Management System. http://www.novell.com/products/edirectory/.
13. Trust Ontology. http://www.mindswap.org/ golbeck/web/trust.daml/, 2006.
14. N. Shadbolt. A Matter of Trust. In *IEEE Intelligent Systems*, pages 2–3, February 2002.
15. Trust in Web Based Social Networks. http://trust.mindswap.org/.
16. Internet X.509 Public Key Infrastructure Certificate and CRL Profile. http://www.ietf.org/rfc/rfc2459.txt.
17. B. You and M.P. Singh. An Evidential Model of Distributed Reputation Management. In *AAMAS-02*, July 2002.
18. P. Zimmermann. PGP User's guide. In *MIT Press*, 1994.

Biography

Latifa Boursas received a diploma degree in Computer Engineering from the Université des Sciences et de la Technologie (USTHB) of Alger, Algeria. She is a member of the MNM team and PhD candidate at the Technische Universität München (TUM), Germany. She is working in the project IntegraTUM at the Leibniz Supercomputing Center that deals with establishing a user-friendly and integrated information and communication infrastructure at TUM. Her research interests include Trust Management and Federated Identity Management in virtual environments.

Helmut Reiser is since 2005 a head of a Network Planning Group at the Leibniz Supercomputing Center in Garching near Munich. He supervises research groups working in Monitoring, Accounting, VO-Management and Middlewares in the German D-Grid Project, performance analysis and end to end monitoring in Geant2 as well as some industry funded research projects. He is member of the DEISA (Distributed European Infrastructure for Supercomputing applications) networking group.

In 2001 he received his PhD from the Ludwig-Maximilian University in Munich which was about security in mobile agent based management systems. He got a diploma degree in computer science from the Technical University Munich and his actual research interests include security and security management, Grid computing and monitoring.