2012 IEEE 11th International Conference on Trust, Security and Privacy in Computing and Communications

Trustworthy Infrastructure Services for a Secure and Privacy-respecting Internet of Things

Dennis Gessner
NEC Laboratories Europe
Heidelberg, Germany
dennis.gessner@neclab.eu

Alexis Olivereau
CEA, LIST
France
alexis.olivereau@cea.fr

Alexander Salinas Segura
University of Wuerzburg
Wuerzburg, Germany
alexander.salinas@uni-wuerzburg.de

Alexandru Serbanati
CATTID - Università Sapienza di Roma
Rome, Italy
a.serbanati@uniroma1.it

***Abstract*— Security is an important cornerstone for the Internet of Things (IoT). Due to the expected pervasion of IoT and its relevance in all fields of human activity, it will likely become a critical asset. Thus, the integrity of data and trust in the services offering the data is crucial. Further, to protect important data and user interests, confidentiality of data and privacy of users must be ensured. Moreover, each request and response in the frame of IoT has to be authenticated in a proper and secure way to ensure accountability and proper operation. Finally, with the usage of IoT for vital functionalities, availability becomes increasingly important, although availability is out of the scope of this document.**

The resolution infrastructure introduced in this work is a crucial component of the overall IoT architecture and most security goals are anchored here. Our suggested architecture ensures privacy and security for the resolution functions and offers as well a basis for other security functionalities needed outside the resolution infrastructure.

IoT; WSN; Security; Privacy

I. INTRODUCTION

In the past, Wireless Sensor Networks (WSNs) were incorporated in vertical applications to satisfy specific network demands in particular domains. For this purpose the sensor network design itself has been developed for a predefined user scenario or application field. Although in most of the cases this approach works properly, the IoT Architecture (IoT-A) project aims at providing interoperability among different WSN- and RFID-based solutions operating in different areas. In order to provide a common IoT substrate and support heterogeneous IoT scenarios, a secure and trustworthy resolution infrastructure is needed to discover and resolve names and identities to addresses used by services.

In contrast to WSNs, we propose a dynamic architecture where a set of IoT resources may only be determined at runtime. That implies that a new level of flexibility needs to be introduced considering both the identification and the resolution of attached IoT-devices or services. This poses a series of challenges to the security of the system and the privacy of users which must be taken into account.

This paper presents the current progress on the security functional components of the IoT-A resolution infrastructure. It describes the ideas, the architecture, the design and multiple implementation options for the security components of the proposed IoT architecture.

II. RELATED WORK

In the nomenclature of access control systems definitions (e.g. [1]) the authorize functionality, realised in this work as AuthZ component (described in section IV.A.), is called Policy Decision Point (PDP), while the calling IoT service resolution component is called Policy Enforcement Point (PEP). These are the basic access control components which must always be present.

Two widespread access control models, which are relevant for this work are role-based access control (RBAC) and attribute-based access control (ABAC). RBAC [2] is a widespread access control model, especially in commercial applications. It maps permissions (i.e. access to resources) not directly to users but to roles. Additionally, it contains a mapping (many to many) from users to roles, defining which users are allowed to take which role. Together with the concept of role hierarchy, RBAC allows the creation of rather compact policies. In many contexts, “real life roles” like job functions can be mapped to policy roles, giving a convenient way of creating roles.

In ABAC access is not granted based on the identity of the user, but on the attributes he is presenting inside the assertion. The ABAC policy maps attributes to permissions (i.e. access to resources). A common example for an authentication attribute is “older than 18 years” allowing access to adults without identifying the user.

In areas where the access policies are not fixed, some component has to offer an interface for changing the policy. This interface is called Policy Administration Point (PAP) [3]. The actual parameters of this interface depend on the policy model and language used in this policy handling component. In this work, we define abstract operations (derived from [4], described in section IV.3.).

As with most networks, the IoT requires in many scenarios that a user passes through an authentication mechanism in order to log onto the network or access any IoT services. The main approaches to user authentication fall into three main types [5], these are:

- knowledge-based
- possession-based, or
- biometric-based

In [6], the authors describe a possible way to hide real-world identities behind pseudonyms.

Different definitions of Trust and Reputation have been given along several research works [7] [8] [9]. However, some authors do not explicitly distinguish between the

978-0-7695-4745-9/12 $26.00 © 2012 IEEE
DOI 10.1109/TrustCom.2012.286

concepts of Trust and Reputation whereas others make a clear separation between them, this work will consider them as different but closely related issues.

Most trust and/or reputation models generally follow five generic steps [10]: gathering behavioural information, scoring and ranking entities (sensors, or even their measurements, in the case of an IoT architecture), entity selection, transaction, and rewarding and punishing entities.

Models of trusting the entities in preparation, so called Pre-Trust of entities, are introduced by EigenTrust [11] or PowerTrust [12]. To identify entities being malicious or not, Regret [13], MTrust [14] or [15] are mentioned, who not only evaluate the entities as service providers, but also as ratings providers. An example of malicious entities, capable of performing Collusion or Sybil attacks is introduced in [16].

III. GOALS AND APPROACH

The IoT-A project has gathered from internal and external stakeholders a set of requirements for what concerns security and privacy. Thus a set of required functionalities has been derived from external stakeholder requirements with a top-down approach. These functionalities have been integrated with another set of functionalities resulting from the analysis of requirements provided by internal stakeholders that are more technical. The result was a comprehensive set of security functionalities that are needed in the IoT.

Some of these functionalities could not be provided by only the two nodes respectively invoking and providing a service and thus needed to rely on other trustworthy components. And, because these functionalities had to be provided for all possible groups of nodes, infrastructure functional components were taken into account. Three criteria were used for identifying these components and assigning functionalities: logical affinity among functionalities, minimization of the number of interfaces (and thus shared information) among components, and minimization of resources requested to IoT nodes.

In dealing with privacy, the concept of identity is crucial for almost all components described in this work. We define identity in the frame of IoT as the whole set of digital information pertaining to a subject (identifier + secret + locator) and allowing him to distinguish himself and to be distinguished from other subjects. An identity can be associated only to one subject, yet a subject can use multiple identities at the same time.

IV. COMPONENT DESCRIPTION

A. Authorization (AuthZ)

1) Access control interface

The authorization component (AuthZ) is a front end for performing access control decisions based on access control policies. This access control decision can be called whenever access to a restricted resource is requested. For example, this function is called from the IoT service resolution component, to check if a user is allowed to discover or perform a lookup on the requested resource. This is an important part of the privacy protection mechanisms.

From an abstract point of view an access control decision can be modelled like this:

$$authZ(s, r, o) \rightarrow \{true, false\} \quad (1)$$

where s is the subject performing the access, r is the resource to be accessed and o is the operation to be performed on the resource. This means, the function authZ evaluates if s is allowed to access r using the operation o. As an example, we consider the UNIX file system. Here, s is a system user, r is a file/ folder on the file system and o is "read", "write" or "execute".

Our AuthZ component offers this functionality in the following form:

Boolean: authorize(Assertion, Resource, ActionType) (2)

It uses the following data type parameters:

- Boolean: the result of the access control decision, with TRUE = PERMIT and FALSE = DENY. In addition, NON-APPLICABLE could be returned. The root policy must be defined in a way that these results are mapped to PERMIT or DENY.
- Assertion: representing information about the accessing user, depending on the scenario. Typical examples are: user ID, certificate, SAML assertion, Kerberos assertion etc.
- Resource: representing the resource to be accessed. Examples for resources are service descriptions (for resolution operations) or services (for service invocations).
- ActionType: representing the operation to be performed on the resource. These operations can be "lookupService" or "discoverService" for the IOT service resolution or information access methods for other services. Examples for the latter one might be "readMaxTemperature" or "resetAirPressureValues" for a weather service.

2) Access Control Models

The access control decision taken by an authorization point can be based on different access control models (RBAC and ABAC, introduced in section II.). These access control models again can be implemented using fine granulated access control policies. In contrast to RBAC, the main advantage of ABAC is a higher level of privacy for the user. However, it also requires a higher trust to the entity creating the assertion (here: the authentication component called AuthN, see section IV.B.). Thus, in scenarios where the AuthN and the AuthZ component are located in different security domains, this fact must be considered in the phase of selecting an access control model.

3) Access Control administration

For very basic nodes in an IoT, the access control policy will mostly be fixed for the whole lifetime of the node. It will be defined during deployment or even design time and installed immutable on the node. In these case changes regarding access rights can only be performed indirectly (and in some cases even not at all). As an example, if the policy allows access for users presenting an assertion signed by a

specific third party, access rights can be changed by changing the set of users allowed to request such an assertion.

Table 1 describes a set of operations to be able to modify existing policies, e.g. called whenever a new service is registered to the IoT resolution infrastructure. In this case the Resource Manager must set the access control rules of this service.

TABLE I. DEFINITION OF OPERATIONS

Operation	Input	Output	Description
setPolicyRoot	Policy		Set up a root policy or policy set
getPolicies		PolicySet	Returns the current root policy set
setPolicy	Policy		Overrides a policy (with the same policy ID)
getPolicy	ID	Policy	Returns a policy with a given policy ID
addPolicy	Policy, ID		Adds a policy to a policy set with a given policy set ID
deletePolicy	ID		Deletes a policy with a given policy ID

4) Relationships with other components

To use the AuthZ component, the AuthN component is required. The authorisation component needs to verify the assertion that was generated by the authentication component. In addition, resolving an identity might be restricted in IoT-scenarios for privacy reasons. In these cases, the resolution component calls AuthZ to check whether the user is allowed to resolve the pseudonym.

B. Authentication (AuthN)

The confirmation of a user or service identity for accessing certain IoT resources is part of the functionalities for which the Authentication (AuthN) component is responsible. It is though also closely related to all the other components and precedes their usage. For example, AuthN approves the stated identity of a subject while AuthZ confers on authenticated subjects the credentials for specific operations.

The operation of the AuthN component and its basic functionality can be stated in the following form:

$$Assertion: authenticate(UserCredential) \quad (3)$$

It uses the following data type parameters:

- Assertion: Data guaranteeing for the occurrence of an authentication of a user client at a particular time using a particular method of authentication. The assertion declares that a specific subject is authenticated by the issuing authority at a given time. It serves as a precondition not only for using the IoT Service Client or the IoT service resolution component but also for the verification of the assertion itself in the AuthN in a later step and the AuthZ.
- UserCredential: Solicited input for the authentication process that must be present in order to perform authentication. The most frequent form is the user/password way of identification, however there exist other forms such as shared key, digital certificate, or biometric credential.

A further functionality of the AuthN component is its role in the bootstrapping of a new node joining the IoT infrastructure. To that respect, the AuthN component is to feature the server-side part of an authentication for network access exchange, often referred to as "authentication server" in network-related specifications. The AuthN authenticates a newly joining node by checking the correctness of the credentials this node supplies. This initial authentication may lead to the establishment of secured contexts between the newly joining node and various entities in its local environment (e.g. neighbour nodes, IoT gateway) as well as between the newly joining node and resolution entities (Resolution Server, Key Exchange and Management component).

C. Identity Management (IM)

The pervasiveness of the Internet of Things raises many privacy issues. The greater threats to privacy come from the possibility to retrieve information about or related to people by using (or subverting the use of) the Internet of Things. The same ability of third parties to know that two entities are exchanging data can be a violation of privacy. Moreover, some services should be accessed on an anonymous basis, while others might require an explicit authentication or authorization of the user. On the other hand there might be services that need to be provided in an anonymous way.

The Identity Management functional component isolates all the functionalities that affect and directly deal with digital identities and, in particular, it is also responsible for creating and integrating new identities in the IoT system. There should be no way to distinguish between root, pseudonym and group identities.

IM is also the component that provides the main support for the satisfaction of privacy and accountability requirements.

Specifically, the following functions are provided.

1) Manage identity lifecycle

A set of processes have to be carried out for synchronizing the infrastructure components and ensuring correct and secure functioning. These processes include associating to a given identity an expiration date, an authentication scheme and related credentials, as well as its access rights.

2) Create root identities

Root identities are digital identities that directly identify a real-world subject. This association is recorded by the component at the time of the creation of the new root identity. For security and privacy reasons the use of this kind of identities should be limited to the request of pseudonyms.

Upon the creation of a root identity, IM will associate real-world subject identities with authentication mechanisms

and credentials, authorization access rights as well as setting up associations in resolution entries.

3) Create pseudonyms

Upon the request of (pseudonym- or root-) authenticated subjects, the IM component creates a new pseudonym that can be used by the subject in order to authenticate without revealing its root identity. Pseudonyms can also be used to provide interoperability between different Identity Frameworks using different algorithms and key lengths. In this case, the IM component will be leveraged in order to obtain an identity that can be authenticated in a different Identity Framework.

As a general rule for pseudonym creation, the new identity must have less access rights and a shorter lifetime. In case the request is for a pseudonym that must work in a different Identification framework, the security level of the new identity must be lower than that of the identity used for requesting the pseudonym.

In order to better protect the privacy of the subjects, the address of the subject must also be changed

4) Create group pseudonyms

There might be structures of subjects that could benefit from having a group identity with associated access rights. Group pseudonyms must be requested by a subject which is going to be responsible for the use of the resulting identity by a set of subjects. The requesting subject will be also in charge of the distribution of the credentials to the nodes, as these subjects are not connected to the IM component at the moment of the request. The access rights will be managed in the same way as those of standard identities.

5) Relationships with other components

IM leverages AuthN component to authenticate the pseudonym requesting subject. IM will use AuthZ in order to 1) learn the access rights of the requesting subject in order to verify the consistency of the request and 2) set the access policy for a new pseudonym. Interaction with AuthN and AuthZ, KEM and TRA is also necessary during the identity lifecycle management. The IM component will interact with the Resolution component when creating a new pseudonym in order to set a new identifier-locator association in order to provide anonymous interactions and the unlinkability feature.

D. Key Exchange and Management (KEM)

The Key Exchange and Management (KEM) component assists IoT peers in the establishment of a secured context between them, when they would be unable to do so if relying on their own abilities. This may involve the provision of adequate key material to the peers, as well as the set-up of interoperability functions between them. The KEM component may also bootstrap assisted security, wherein the security level of communications between a set of peers is increased through the involvement of one or more additional peers. These latter may be only involved in the (costly) operation of secure context establishment, or may be actively taking part to the security of a communication channel, providing hop-by-hop security when available.

Location of the KEM component within the resolution infrastructure is judicious for multiple reasons. The first reason is that the establishment of a secured context between two nodes is generally immediately preceded by resolution of one node by the other. Hence the key establishment operation can reasonably be bootstrapped by the resolution one. This is an efficiency reason, whose relevance is further increased if the initial resolution operation includes provision of information about the to-be-established communication (e.g. sensitivity). Another reason is that the resolution system may have learnt relevant information about the nodes' supported identification mechanisms, or about the nodes' capabilities. This may occur as part of a dedicated operation upon node's registration, but it may also be logically induced from node's identifier type (e.g. node using an identifier implicitly classifying it as a low-resource node).

It has to be noted that, although part of the resolution infrastructure, the KEM component may be used without relying on interactions with dedicated resolution entities in this infrastructure. It is possible that a set of nodes determine by themselves (or at least, without using the resolution components) that they are not interoperable and that they have to bootstrap an interoperability system for establishing secure communications between them. In that situation, the set of nodes may directly contact the KEM component to access the services it offers.

Whether invoked by other resolution entities or by the IoT nodes themselves, the KEM component operation is as follows:

1) Request. In addition to the requesting node's and targets' identifiers and identification means, various parameters can be carried within the request that pertain to the required security level, such as:

- Type of authentication. The requesting node may wish to authenticate its peer(s) using an end-to-end scheme (in which case the KEM will make in sort to provide all peers with the appropriate key material, allowing them to carry out end-to-end authentication) or may accept to rely on hop-by-hop authentication if it turns out for example, that a chain of trust exists between the requesting node and its target node(s).
- Required security level and key purpose, especially indicating whether an Authenticated Key Exchange (AKE) protocol will be run between the communicating nodes in a subsequent step (in that case the requesting nodes expects that the KEM puts in place credentials bootstrapping AKE protocol) or whether applicative keys are requested from the KEM.

2) Action. As a result of the key enablement request, the KEM takes appropriate action through interaction with (1) the requesting node and/or (2) designated target node(s) and/or (3) assisting entities, so as to ensure the subsequent proper setup of a security relationship between the requesting node and the target node(s), possibly through the assistance of the assisting entities (assisted keying, on-path gateways, interoperability servers, group security servers).

Interactions (1) and (2) generally consist of secure key delivery. The keys must be delivered along with a key scope,

which represents the identities[1] of all peers to which the same key material will be disclosed. An indication of the key purpose may be included too, as well as a globally unique key name.

The latter category of assisting entities may involve complex scenarios depending on trust relationships between nodes, network topologies, or identification models used by requesting and target nodes. For example, a collaborative security establishment scheme can be bootstrapped between a constrained requesting node and local trusted less constrained nodes. If both requesting node and target node are connected through gateways to the IoT, a secured tunnel can be dynamically setup between their respective serving gateways. Finally, if the requesting node and the target node do not share the same identity model (e.g. Host Identity Protocol (HIP) vs. non-HIP), an interoperability server can be dynamically configured.

E. Trust and Reputation Architecture (TRA)

1) Generic steps and operations

The first component of a standard trust and/or reputation model, gathering information, should be responsible for collecting behavioural information about the entities in the system to be used to determine how trustworthy an entity is (either on an absolute scale or relative to other entities). This information might come from several sources such as direct experiences with the targeted entity, neighbours, acquaintances, belonging group or organization, and even witnesses. Pre-trusted entities would be also acceptable as it already happens in existing models (see section II.).

In our work, a sensor might provide bogus or erroneous information either due to explicit misbehaviour, or just as a result of a malfunctioning device. At this point, several specifically applicable security threats [17] [18] may be taken into consideration.

Once an entity's transaction history has been collected and weighted, a trust and/or reputation score should be computed for that entity with fuzzy logic, bayesian networks, analytic expressions or bio-inspired algorithms. Taking the constrained node resources into consideration, the reputation computation engine to be deployed has to be light, scalable and robust, amongst other features.

In the end, a local or global trust and/or reputation rating is done, used in order to help an entity to decide which entity to interact with. This can be modelled as a binary value (e.g. trusted, untrusted), a scaled integer (e.g. 1, 2, 3,…), an element from a set of linguistic labels (e.g. {"very trustworthy", "trustworthy", "untrustworthy", "very untrustworthy"}), a value within a continuous interval (e.g. [0, 1]), etc.

Trust and/or reputation mechanisms could consider the behaviour of an entity as a whole, while others distinguish among the different services offered, by assigning a different rating related to each service. An incorrect handling of this issue might lead to critical risks. For instance, one single sensor might be able to measure different things or could be seen as a "service provider" offering several different "services".

Having selected an entity to interact with, the transaction itself would be carried out between both entities, giving a certain service or good as a result. In our particular case, the requester would receive the measurement from the selected sensor. Finally, after receiving the requested service, the client entity should assess that transaction so it can reward or punish the entity, providing that service. In our opinion, this is a very important component in a trust and/or reputation model, since the accuracy of trust and/or reputation ratings highly depend on how fair an entity is evaluated from a given service.

We suggest generic operations aimed to retrieve reputation information about an entity and to provide feedback or recommendation about such entity, respectively. These two operations are:

RepBundle reqRepInfo (Subject subj, Context cont) (4)

Rep provideRepInfo (Subject subj, Context cont, ScoreType score, DateType timestamp) (5)

(4) is invocated by a given remote entity to request reputation information about another entity. As input parameters, a unique identifier for the remote entity (subject), as well as the concrete context is given.

In turn, (5) is invocated by a given remote entity to provide reputation information (recommendations or feedback) about another entity. As input parameters, a unique identifier for the entity to be assessed (subject), as well as the concrete context, the given score and a timestamp are given. As a result, the corresponding reputation element is provided.

2) Design recommendations & security threats

In the previous section we described the components a standard trust and/or reputation model should have. We consider some recommendations as more critical and unavoidable in any trust and reputation mechanism, whereas others are more susceptible to each specific environment.

- As a specific application of privacy, entities' anonymity should be considered. However, they should not be indistinguishable and unlinkable.
- During computing the trust and/or reputation values, the more recent a transaction is, the more weight it should have.
- An entity may evaluate a performed transaction alone or in combination with other parties.
- There must be the possibility to revoke one or more untrusted entities, even if the entity had a very good trust and/or reputation in the past.
- Initial trust and/or reputation level of new nodes needs to be considered, thus new nodes will not automatically be selected.
- To avoid misbehaviour of trusted entities, the computation always needs to be up to date.

[1] Note that if one considers a key disclosed to a pair of nodes A and B, the key scope for A will consist of (A, IDBtoA) where IDBtoA is the identity of B as seen by A. Likewise, the key scope for B will consist of (B, IDAtoB). For privacy reasons, both A and IDAtoB and B and IDBtoA may be different.

- Newcomers should not have more communication privileges than non malicious remaining nodes in the network.
- Every entity should receive a different rating depending on the type of service it is providing.
- Trust and/or reputation models should take care about the overhead they introduce in the system.
- Finally, the more relevant a transaction is, the higher should his gain or loose of trust and/or reputation value needs to be.

3) Relationships with other components

The TRA component needs to uniquely identify each subject (sensor) to interact with, both when requesting recommendations about other entities and when asking for a concrete service (measurement). Additionally there is the necessity of a queried entity to check whether a requestor might have enough privileges or not to actually query for certain reputation information.

When creating a new pseudonym, the IM component also has to create a new address in order to provide anonymous interactions and the unlinkability feature, nevertheless the new address needs to fulfil all features of the TRA component.

Finally, when exchanging reputation material or information, such communication must be secured, thus, the KEM component needs to provide a cryptographically secure information flow.

V. CONCLUSIONS AND OUTLOOK

This paper proposes a comprehensive set of trust-enhancing security functional components for the resolution infrastructure as a crucial part of the Internet of Things.

A preliminary requirement analysis and critical control points assessment lead to the introduced components in order to cover not only basic IoT resource access control (AuthZ & AuthN), but also essential functions such as identity management (IM), key exchange and management (KEM) and trust and reputation management (TRA). This component composition with its interdependencies provides compulsory mechanisms for securing communication between subjects to guarantee an inviolable interaction and therefore incorporates dedicated aspects of data integrity and confidentiality, service trust and privacy of users to name but a few. As a generally valid guideline how to deal with security and privacy in the IoT, the presented security functional components pave the way for future IoT applications.

However, as an initial step towards a fully-fledged IoT security model further improvements in later stages are still necessary. This implies an extension of security and privacy related mechanisms via a higher degree of level of detail for certain components through to an implementation of a prototype.

ACKNOWLEDGMENT

THE WORK PUBLISHED IN THIS PAPER IS (PARTLY) FUNDED BY THE EUROPEAN COMMISSION THROUGH THE IOT-A-PROJECT. IT DOES NOT REPRESENT THE VIEW OF EC OR THE IOT-A CONSORTIUM, AND AUTHORS ARE SOLELY RESPONSIBLE FOR THE PAPER'S CONTENT.

REFERENCES

[1] J.Vollbrecht, P. Calhoun, S.Farrell, L.Gommans, G.Gross, and B.de Bruijn, C.de Laat, M. Holdrege, and D. Spence: AAA Authorization Framework, RFC 2904 (Informational)

[2] D.F. Ferraiolo, D.R Kuhn: Role-Based Access Control. 15th National Computer Security Conference. pp. 554–563, October 1992.

[3] Tim Moses (Ed.): eXtensible Access Control Markup Language (XACML) Version 2.0, OASIS Standard, 1 Feb 2005, http://docs.oasis-open.org/xacml/2.0/access_control-xacml-2.0-core-spec-os.pdf

[4] X A C M Light, http://xacmllight.sourceforge.net/

[5] Menkus, B. (1988) "Understanding the Use of Passwords," Computers and Security, 7(2), pp. 132-136.

[6] A. Singh, L. Liu, TrustMe: Anonymous Management of Trust Relationships in Decentralized P2P Systems, IEEE International Conference on Peer-to-Peer Computing, 2003, pp. 142–149.

[7] A. Abdul-Rahman, S. Hailes, Supporting Trust in Virtual Communities, Proceedings of the 33rd Hawaii International Conference on System Sciences. Hawaii, USA, 2000.

[8] D. Gambetta, Can we trust trust? in: D. Gambetta (Ed.), Trust: Making and Breaking Cooperative Relations, 2000, pp. 213–237, Published Online, Ch. 13.

[9] Marsh, S.P., Apr. 1994. Formalising trust as a computational concept. Ph.D. thesis, Department of Computing Science and Mathematics, University of Stirling.

[10] S. Marti, H. Garcia-Molina, Taxonomy of trust: categorizing P2P reputation systems, Computer Networks 50 (4) (Mar. 2006) 472–484.

[11] S. Kamvar, M. Schlosser, H. Garcia-Molina, The EigenTrust Algorithm for Reputation Management in P2P Networks, Proc. of the International World Wide Web Conference (WWW). Budapest, Hungary, May 2003.

[12] R. Zhou, K. Hwang, PowerTrust: A Robust and Scalable Reputation System for Trusted Peer-to-Peer Computing, Transactions on Parallel and Distributed Systems, 2007.

[13] J. Sabater, C. Sierra, REGRET: reputation in gregarious societies, in: J.P. Müller, E. Andre, S. Sen, C. Frasson (Eds.), Proceedings of the Fifth International Conference on Autonomous Agents, ACM Press, Montreal, Canada, 2001, pp. 194–195.

[14] S. Songsiri, MTrust: a reputation-based trust model for a mobile agent system, Autonomic and Trusted Computing. No. 4158 in LNCS. Third International Conference, ATC 2006, Springer, Wuhan, China, Sep. 2006, pp. 374–385.

[15] F. Yu, H. Zhang, F. Yan, S.Gao,An improved global trust value computing method in P2P system, Autonomic and Trusted Computing. No. 4158 in LNCS. Third International Conference, ATC 2006, Springer, Wuhan, China, Sep. 2006, pp. 258–267.

[16] J.R. Douceur, J.S. Donath, The sybil attack, Proceedings for the 1st International Workshop on Peer-to-Peer Systems (IPTPS '02), 2002, pp. 251–260.

[17] F. Gómez Mármol, G. Martínez Pérez, Security threats scenarios in trust and reputation models for distributed systems, Elsevier Computers & Security 28 (7) (2009) 545–556.

[18] Y. Sun, Z. Han, K. Liu, Defense of trust management vulnerabilities in distributed networks, IEEE Communications Magazine 46 (2) (Feb. 2008) 112–119.