2004

Modelling trust relationships in distributed environments

Weiliang Zhao
University of Western Sydney, wzhao@uow.edu.au

Vijay Varadharajan
University of Western Sydney

George Bryan
University of Western Sydney

http://ro.uow.edu.au/engpapers/5213

Publication Details

Zhao, W., Varadharajan, V. & Bryan, G. (2004). Modelling trust relationships in distributed environments. International Conference on Trust and Privacy in Digital Business (pp. 40-49). Berlin Heidelberg: Springer.

Modelling Trust Relationships in Distributed Environments

Weiliang Zhao[1], Vijay Varadharajan[1,2], and George Bryan[1]

[1] Centre for Advanced Systems Engineering
University of Western Sydney
Locked Bag 1797
Penrith South DC, NSW 1797, Australia
{wzhao,g.bryan}@cit.uws.edu.au
[2] Department of Computing
Macquarie University
NSW 2109, Australia
vijay@ics.mq.edu.au

Abstract. Trust management and trustworthy computing are becoming increasingly significant at present. Over the recent years there have been several research works that have addressed the issue of trust management in distributed systems. However a clear and comprehensive definition that can be used to capture a range of commonly understood notions of trust is still lacking. In this paper, we give a formal definition of trust relationship with a strict mathematical structure that can not only reflect many of the commonly used extreme notions of trust but also provides a taxonomy framework where a range of useful trust relationships can be expressed and compared. Then we show how the proposed structure can be used to analyze both commonly used and some unique trust notions that arise in distributed environments. This proposed trust structure is currently being used in the development of the overall methodology of life cycle of trust relationships in distributed information systems.

1 Introduction

The concept of trust has been used and studied in social science for a long time [1, 2]. Trust was originally used in human and social issues in day-life relationships, laws, regulations and policies. In the computing world, the trust was originally used in the context of trusted computing such as trusted system, trusted hardware and trusted software [3]. Recently, trust has been used in the context of trust management in distributed computing [4–7]. When the Internet and web technologies are broadly and increasingly used in daily life for electronic commerce, trust becomes a very hot topic [8, 9]. The trust between customers and e-vendors includes not only technical aspects but also social aspects. In this paper, we will provide our definition of trust relationship. Most of the issues relating to social aspects of trust is beyond the scope of this paper, but we hope that our general definition of trust relationship can cover both aspects. The trust relationships of involved entities or computing components in distributed computing are our major concern.

XML-based Web Services technologies have been rapidly evolving since 1999. Web Services technologies address the challenges of distributed computing and B2B integration. There are huge number of service oriented applications on the Internet and they are coupled loosely. Web Services technologies target at loosely-coupled, language-neutral and platform-independent way of linking applications for business process automation within organizations, across enterprizes, and across the Internet. There is no centralized control and the users are not all predetermined. Normally, the computing components involved in a e-service can belong to different security domains and there is no common trusted authority for the involved entities. How to define/model trust relationships between computing components is an important and challenging issue in the design of web services. The draft of WS-Trust was proposed in 2002 [10]. Unfortunately, the current WS-Trust only touches the issue of trusted message exchange and has not provided more details for dealing with trust relationships.

Many researchers have recognized the trust management as a distinct and important component of security in distributed systems. Several automated trust management systems have been proposed such as PolicyMaker[4], KeyNote[5, 6], and REFEREE [7]. In all these trust management systems, trust and its related concepts are assumed in a specific way relating to the specific topics. There is no consensus on the definition of trust. In PolicyMaker and KeyNote, M. Blaze *et al* provided clear definition of trust management system and there are many clues to understand what is trust but they did not comment on the concept of trust directly. In REFEREE, Y. H. Chu *et al* described trust as "to trust is to undertake a potentially dangerous operation knowing that it is potentially dangerous". Tyrone *et al* [11] gave a definition of trust as "the firm belief in the competence of an entity to act dependably, securely, and reliably within a specified context". Y. H. Chu *et al* and Tyrone *et al* talked about trust in a kind of general terms, however trust is difficult to express without a strict mathematical structure. In PolicyMaker, KeyNote and REFEREE, a new trust management layer has been successfully built but the concept of trust and how to model trust has not been considered carefully. It is necessary to have a solid understanding of the concept of trust relationship and to develop a powerful set of tools to model the trust relationships for trust management in distributed information systems.

The starting point of this research is trust in the context of distributed environments. Here we have not separated the traditional distributed computing and the Web Services. Web Services are included when we talk about distributed computing for the consideration of trust issues.

The rest of the paper is structured as follows. In section 2, we give the definition of trust relationship and discuss some extreme cases. In section 3, we give a series of definitions, propositions and operations about trust relationships. The mathematical properties of trust relationships are embedded in these definitions, propositions and operations. In section 4, we provide two scenario examples of trust relationships and we give some analysis of trust relationships using the definitions, propositions and operations in section 3. In section 5, we provide concluding remarks.

2 Definition of Trust Relationship

Most of the researchers agree that a trust relationship is the relationship between a set of trusters and a set of trustees in a specified context, but it is not clear enough, especially when it is used in the computing world. There is a need to convert the generally used terms into strict mathematical structure in algorithms of real systems. In this paper, we will provide our definition of trust relationship with a strict mathematical structure.

In trust management of distributed information systems, we believe that the definition of trust should have the following characteristics:

- The definition of trust is unique and can be used for different computing purposes.
- The definition of trust has strong expressive power and makes the system as simple as possible.
- The definition of trust has a strict mathematical structure.
- The definition of trust provides the solid foundation for discussing the properties of trust relationships.
- The definition of trust follows hard security mechanisms.

Hard security assumes complete certainty and it allows complete access or no access at all. Here we only model the static status of trust in distributed environments.

We believe that it is not enough to understand trust as a simple bilateral relation between trusters and trustees. The whole syntax of trust relationship should be "under a set of specified conditions, a set of trusters trust that a set of trustees have a set of specified properties (the set of trustees will/can perform a set of actions or have a set of attributes)". The definition is expressed as follows:

Definition 1 *A trust relationship is a four-tuple $T =< R,\ E,\ C,\ P >$ where:*

- *R is the set of trusters. It contains all the involved trusters. It can not be empty.*
- *E is the set of trustees. It contains all the involved trustees. It can not be empty.*
- *C is the set of conditions. It contains all conditions (requirements) for the current trust relationship. Normally, trust relationship has some specified conditions. If there is no condition, the condition set is empty.*
- *P is the set of properties. The property set describes the actions or attributes of the trustees. It can not be empty. The property set can be divided into two sub sets:*
  - *Action set: the set of actions what trusters trust that trustees will/can perform.*
  - *Attribute set: the set of attributes what trusters trust that trustees have.*

Anywhere, a trust relationship must be used with full syntax(four-tuple $< R, E, C, P >$. Trust relationship T means that under the condition set C, truster set R trust that trustee set E have property set P. There are some extreme cases of the trust relationship when some involved sets included nothing(empty set) or anything(whole set of possible entities). The extreme cases have special meanings and are crucial in the understanding of the definition of trust relationship. These extreme cases will play important roles in the real world. The followings are the five extreme cases of trust relationship:

1. R is ANY. Truster set includes all possible entities. All possible entities trust that the set of trustees E have the set of properties P under the set of conditions C.

2. E is ANY. Trustee set includes all possible entities. All possible entities can be trusted to have the set of properties P by the set of trusters R under the set of conditions C.
3. C is $EMPTY$. There is no condition in the trust relationship. The set of trusters R trust that the set of trustees E have the set of properties P without any condition.
4. P is ANY. The property of the trustee can be anything. The set of trusters R trust that the set of trustees E have all possible properties under the set of conditions C.
5. C is $EMPTY$ and P is ANY. The set of trusters R trust that the set of trustees E have all possible properties without any condition. This case happens when the set of trusters R trust the set of trustees E by default.

When the full syntax of the trust relationship is not used, trust relationship is easily misunderstood. Normally, there are many implicit assumptions and some parts of full syntax are usually omitted. When we analyze the true meaning of a trust relationship, the full syntax must be recovered. Our definition of the trust relationship has strict mathematical structure with the full syntax in any case. There is no confusion when the full syntax trust relationship is used in any information system.

It is straightforward to use the set of conditions in the definition of trust relationship. When a trust relationship is used, trusters, trustees and properties are normally involved individually. The trust relationship can always be evaluated based on one truster, one trustee and one property. In our definition of trust relationship, the trusters, trustees and properties turn up as sets are based on the following concerns (1) The concept of security domain is broadly used and related technologies are quite mature. The role-based access control is broadly used and well understood by programmers and business people. When a set of trusters, a set of trustees and a set of properties are used in the definition of trust relationship, the similar ideas in security domain and role-based access control can be employed easily. It is convenient to define some abstraction characteristics based on a group of trusters, a group of trustees and a group of properties. We hope that a set of trusters, a set of trustees and a set of properties in the definition of the trust relationship have better abstraction and it is easier to use the definition. (2) The set theory can provide formal mathematical notion and handy tools to discuss the relationships of sets. (3) An individual truster (or trustee, or property) is a special case of the set of trusters (or trustees, or properties). (4) It is convenient to discuss special cases of trust relationship when truster (or trustee, or property) is anyone.

3 Mathematical Properties of Trust Relationships

In this paper, we will discuss the mathematical properties of trust relationship based on our strict definition of trust relationship. The trust relationship has a full syntax with truster set, trustee set, condition set and property set. It is incorrect to only talk about the trust relationship between trusters and trustees without mention of the condition set and property set. The discussions of properties of trust relationship should be based on the full syntax of trust relationship in its definition. In the following part of this section, we will give some definitions, propositions and operations related to trust relationships. The mathematical properties of trust relationships are embedded in these definitions, propositions and operations. These mathematical properties focus on some relations of trust relationships and they will be used as tools in the analysis and design of trust relationships in real systems.

From the nature of trust relationship and its mathematical structure, some new trust relationships can be derived based on the existing trust relationships. In the follows, we will define the operations of using two existing trust relationships to generate a new trust relationship under specific constraints and operations of decomposing one existing trust relationship into two new trust relationships under specific constraints.

Operation 1 *Let* $T_1 = (R_1,\ E_1,\ C_1,\ P_1)$ *and* $T_2 = (R_2,\ E_2,\ C_2,\ P_2)$. *There is a set* $T = (R_1 \cap R_2,\ E_1 \cap E_2,\ C_1 \cup C_2,\ P_1 \cup P_2)$. *If* $R_1 \cap R_2 = \emptyset$ *or* $E_1 \cap E_2 = \emptyset$, $T = \emptyset$.

If $R_1 = R_2$ and $E_1 = E_2$, the operation becomes:

Operation 1A *Let* $T_1 = (R,\ E,\ C_1,\ P_1)$ *and* $T_2 = (R,\ E,\ C_2,\ P_2)$. *There is a set* $T = (R,\ E,\ C_1 \cup C_2,\ P_1 \cup P_2)$.

If $R_1 = R_2,\ E_1 = E_2$ and $C_1 = C_2$, the operation becomes:

Operation 1B *Let* $T_1 = (R,\ E,\ C,\ P_1)$ *and* $T_2 = (R,\ E,\ C,\ P_2)$. *Then there is a set* $T = (R,\ E,\ C,\ P_1 \cup P_2)$.

Operation 2 *Let* $T_1 = (R_1,\ E_1,\ C,\ P)$ *and* $T_2 = (R_2,\ E_2,\ C,\ P)$. *There is a set* $T = (R_1 \cup R_2,\ E_1 \cap E_2,\ C,\ P)$.

If $E_1 = E_2$, the operation becomes:

Operation 2A *Let* $T_1 = (R_1,\ E,\ C,\ P)$ *and* $T_2 = (R_2,\ E,\ C,\ P)$. *There is a set* $T = (R_1 \cup R_2,\ E,\ C,\ P)$.

Operation 3 *Let* $T_1 = (R_1,\ E_1,\ C,\ P)$ *and* $T_2 = (R_2,\ E_2,\ C,\ P)$. *There is a set* $T = (R_1 \cap R_2,\ E_1 \cup E_2,\ C,\ P)$.

If $R_1 = R_2$, the operation becomes:

Operation 3A *Let* $T_1 = (R,\ E_1,\ C,\ P)$ *and* $T_2 = (R,\ E_2,\ C,\ P)$. *There is a set* $T = (R,\ E_1 \cup E_2,\ C,\ P)$.

Operation 4 *Let* $T =< R,\ E,\ C,\ P >$. *If there are* $R_1,\ R_2$ *and* $R = R_1 \cup R_2$, *then there are trust relationships* $T_1 =< R_1,\ E,\ C,\ P >$ *and* $T_2 =< R_2,\ E,\ C,\ P >$.

Operation 5 *Let* $T =< R,\ E,\ C,\ P >$. *If there are* $E_1,\ E_2$ *and* $E = E_1 \cup E_2$, *then there are trust relationships* $T_1 =< R,\ E_1,\ C,\ P >$ *and* $T_2 =< R,\ E_2,\ C,\ P >$.

Operation 6 *Let* $T =< R,\ E,\ C,\ P >$. *If there are* $P_1,\ P_2$ *and* $P = P_1 \cup P_2$, *then there are trust relationships* $T_1 =< R,\ E,\ C,\ P_1 >$ *and* $T_2 =< R,\ E,\ C,\ P_2 >$.

This operation has the following special case:

Operation 6A *Let* $T =< R,\ E,\ C,\ P >$*. If there are* $P_1,\ P_2,\ C_1, C_2$ *and* $P = P_1 \cup P_2,\ C = C_1 \cup C_2$*, then there are trust relationships* $T_1 =< R,\ E,\ C_1,\ P_1 >$ *and* $T_2 =< R,\ E,\ C_2,\ P_2 >$*.*

All operations can be used to generate new trust relationships from the existing trust relationships under some specific constrains. The **Operation 1** deals with any two trust relationships and a new trust relationship is possibly generated(if the result is not $\emptyset$). The **Operation 1A, 1B, 2A, 3A** deal with how to use two trust relationships to generate one trust relationship under some specific constraints. The **Operation 4, 5, 6 and 6A** deal with how to decompose one trust relationship into two trust relationships under some specific constraints. **Operation 1A** and **Operation 6A** are inverse operations. **Operation 1B** and **Operation 6** are inverse operations. **Operation 2A** and **Operation 4** are inverse operations. **Operation 3A** and **Operation 5** are inverse operations.

In the following part of this section, we will focus on the relation of trust relationships, especially we will discuss and define the equivalent, primitive, derived, direct redundant and alternate trust relationships. We will classify the direct redundant trust relationships into different types as well.

Definition 2 *Let* $T_1 =< R_1,\ E_1,\ C_1,\ P_1 >$ *and* $T_2 =< R_2,\ E_2,\ C_2,\ P_2 >$*. If and only if* $R_1 = R_2$ *and* $E_1 = E_2$ *and* $C_1 = C_2$ *and* $P_1 = P_2$*, then* T_1 *and* T_2 *are equivalent, in symbols:*

$$T_1 = T_2 \iff R_1 = R_2 \text{ and } E_1 = E_2 \text{ and } C_1 = C_2 \text{ and } P_1 = P_2$$

Definition 3 *If a trust relationship can not be derived from other existing trust relationships, the trust relationship is a primitive trust relationship.*

Definition 4 *If a trust relationship can be derived from other existing trust relationships, the trust relationship is a derived trust relationship.*

Note: Trust relationships are predefined in information systems. A derived trust relationship is always related to one or more other trust relationships. For an independent trust relationship, it is meaningless to judge it as a derived trust relationship or not.

Proposition 1 *If a derived trust relationship exists, there is information redundancy.*

Proof. When the derived trust relationship is moved out of the system, the information of the derived trust relationship has not been lost. The derived trust relationship can be built when it is necessary. From the view point of information, there is redundancy.

Definition 5 *Let* $T =< R,\ E,\ C,\ P >$*. If there is trust relationship* $T' =< R',\ E',\ C',\ P' >$ *and* $T \neq T',\ R \subseteq R',\ E \subseteq E',\ C \supseteq C',\ P \subseteq P'$*.* T *is a direct redundant trust relationship.*

In the following part of this section, we discuss several special cases of direct redundant trust relationships based on the single tuple of trust relationship. We believe that these special cases play important roles in the analysis and design of trust relationships in information systems.

Direct Redundancy Type 1 ***: DLR-redundant trust relationship***
Let $T =< R, E, C, P >$. *If and only if there is a trust relationship* $T' =< R', E, C, P >$ *and* $R' \supset R$, T *is a DLR-redundant trust relationship.*

T is DLR-redundant trust relationship means that there is another trust relationship with super set of trusters and all other tuples are same as peers in T.

Direct Redundancy Type 2 ***: DLE-redundant trust relationship***
Let $T =< R, E, C, P >$. *If and only if there is a trust relationship* $T' =< R, E', C, P >$ *and* $E' \supset E$, T *is a DLE-redundant trust relationship.*

T is DLE-redundant trust relationship means that there is another trust relationship with super set of trustees and all other tuples are same as peers in T.

Direct Redundancy Type 3 ***: DMC-redundant trust relationship***
Let $T =< R, E, C, P >$. *If and only if there is an alternate trust relationship* $T' =< R, E, C', P >$ *and* $C' \subset C$, T *is a DMC-redundant trust relationship.*

T is DMC-redundant trust relationship means that there is another trust relationship with sub set of conditions and all other tuples are same as peers in T.

Direct Redundancy Type 4 ***: DLP-redundant trust relationship***
Let $T =< R, E, C, P >$. *If and only if there is a trust relationship* $T' =< R, E, C, P' >$ *and* $P' \supset P$, T *is a DLP-redundant trust relationship.*

T is DLP-redundant trust relationship means that there is another trust relationship with super set of properties and all other tuples are same as peers in T.

Definition 6 *Let* $T =< R, E, C, P >$, $T' =< R, E, C', P >$ *and* $C \neq C'$. T *and* T' *are alternate trust relationships of each other.*

An alternate trust relationship means that there is an alternate condition set for the same truster set, trustee set and property set. Perhaps, there are multiple alternate trust relationships. In distributed computing, multiple mechanisms and multiple choices are necessary in many situations and it is the main reason why we define and discuss alternate trust relationship here.

Proposition 2 *If* T *is a DMC-redundant trust relationship, there is one or more than one alternate trust relationships which are not DMC-redundant trust relationship.*

Proof. If T is a DMC-redundant trust relationship, there is $T' =< R, E, C', P >$ and $C' \subset C$. T' is an alternate trust relationship of T. If T' is not DMC-redundant trust relationship, the proposition is proved. If T' is a DMC-redundant trust relationship, the next T'' can be found, $T'' =< R, E, C'', P >$ with $C'' \subset C'$. Such a process will continue until the set of conditions includes minimum number of conditions. In every turn of the process, one or more conditions are removed from the condition set. Because C contains limited conditions, the process can finish when no condition can be removed from the condition set. The final set of conditions is C^f. $T^f =< R, E, C^f, P >$ is an alternate trust relationship with non-redundant conditions.

A DMC-redundant trust relationship may have multiple alternate trust relationships with different sets of non-redundant conditions.

4 Scenario Examples of Trust Relationships

In this section, we make up two scenarios for discussing trust relationships in the real world. We hope that these examples can be helpful in understanding the definition of trust relationship and mathematical properties of trust relationships expressed in section 2 and section 3.

Scenario 1: When people want to change their names, they need to apply to a specific organization (In Australia, the organization is the Registry of Birth Deaths & Marriages). The officers in the organization and the requesters are involved in this scenario. Using the full syntax of our definition of trust relationship, some trust relationships may be modelled as follows:

TS1- 1 *Officers trust requesters if requesters have their Birth Certificate & Driver's Licence that requesters have the right for the change.*

TS1- 2 *Officers trust requesters if requesters have their Citizenship Certificate & Driver's Licence that requesters have the right for the change.*

TS1- 3 *Officers trust requesters if requesters have their Birth Certificate & Citizenship Certificate & Driver's Licence that requesters have the right for the change.*

If **TS1-1**, **TS1-2** and **TS1-3** are all the trust relationships in this information system, based on the definitions and operations in section 3, we can have the following analysis:

- **TS1-1** and **TS1-2** are primitive trust relationships.
- **TS1-1** and **TS1-2** are alternate trust relationships of each other.
- **TS1-3** is a derived trust relationship which can be derived by **Operation 1A** with **TS1-1** and **TS1-2**.
- **TS1-3** is a DMC-redundant trust relationship and it should be removed out of the system.

Scenario 2: An online e-commerce service is called FlightServ which can provide flight booking and travel deals. FlightServ is designed based on the new technologies of web services. FlightServ connects with customers, airlines, hotels and credit card services (some of them maybe web services). The whole system could be very complicated, but we only consider some of trust relationships in the system. In the system, customers are classified into normal flyers and frequent flyers. Originally, some trust relationships are modelled as:

TS2- 1 *Airlines trust normal flyers if they have address details & confirmed credit card information that normal flyers can make their airline bookings.*

TS2- 2 *Airlines trust frequent flyers with no condition that frequent flyers can make their airline bookings.*

TS2- 3 *Hotels trust normal flyers if they have address details & confirmed credit card information that normal flyers can make their hotels booking.*

TS2- 4 *Hotels trust frequent flyers if they have address details & confirmed credit card information that frequent flyers can make their hotels booking.*

TS2- 5 *Credit card services are trusted by all possible entities without any condition that the credit card services will give the correct evaluation of credit card information.*

TS2- 6 *Credit card services are trusted by all possible entities without any condition that the credit card services will keep the privacy of credit card information.*

For the above trust relationships in the system, based on definitions and operations in section 3, we have the following analysis:

- All above trust relationships are primitive.
- Using the **Operation 3A**, trust relationships **TS2-3** and **TS2-4** can be merged to a new trust relationship **TS2-(3)(4)**: "Hotels trust customers if they have address details & confirmed credit card information that customers can make their hotels booking". If **TS2-(3)(4)** has been defined in the system, **TS2-3** and **TS2-4** becomes DLE-redundant trust relationships and will be removed out of the system.
- Using the **Operation 1B**, trust relationships **TS2-5** and **TS2-6** can be merged to a new trust relationship **TS2-(5)(6)**: "Credit card services are trusted by all possible entities without any condition that the credit card services will give the correct evaluation of credit card information & the credit card services will keep the privacy of credit card information". If **TS2-(5)(6)** has been defined in the system, **TS2-5** and **TS2-6** becomes DLP-redundant trust relationships and will be removed out of the system.

Obviously, the definition of trust relationship in section 2 and the mathematical properties of trust relationships in section 3 provide terminologies and helpful tools in the analysis of the two scenarios. In the analysis of the two scenarios, we only employ some definitions, propositions and operations expressed in section 3. We hope that these examples can provide a general picture for the usage of the definitions, propositions and operations. In these two scenarios, we only choose some trust relationships as examples and there are more trust relationships. The systematic methodologies and strategies for modelling trust relationships are beyond the scope of this paper as well and will be discussed elsewhere.

5 Concluding Remarks

The definition of the trust relationship provided in this paper has a strict mathematical structure and broad expressive power. The definition is suitable for any computing purpose. The mathematical properties of trust relationships are shown in a series of definitions, propositions and operations. We believe that these definitions and mathematical properties of trust relationships provide useful tools for enabling the analysis, design and implementation of trust in distributed environments.

This research only provides a starting point for the analysis and design of trust relationships in distributed information systems. How to model trust relationships in distributed information systems and how to merge the trust relationships into the overall distributed information systems provides lots of challenges for further research. We believe that our definition of trust relationship and the associated mathematical properties described in section 3 could be used as helpful tools to model the trust relationships. The definitions and operations in section 3 provide some starting points and tools for the analysis and design of the trust relationships in a system. We are currently working on using the proposed definition of trust relationship and mathematical properties of trust relationships to develop a methodology for modelling trust in distributed systems. This involves several stages such as extracting trust requirements in system, identifying possible trust relationships from trust requirements, choosing the whole set of trust relationships from possible trust relationships and implementing and maintaining trust relationships in systems. We will describe them in details in a separate paper.

References

1. M. Deutsch, “Cooperation and trust:some theoretical notes”, Nebraska Symposium on Motivation, Nebraska University Press, 1962.
2. D. Gambetta, “Can we trust trust?”, In Trust:Making and Breaking Cooperative Relations, Basil Blackwell, Oxford, pp.213-237, 1990.
3. J. Landauer, T. Redmond, et al. “Formal policies for trusted processes”, Proceedings of the Computer Security Foundations Workshop II, 1989.
4. M. Blaze, J. Feigenbaum, and J. Lacy, “Decentralized trust management”, Proceedings of the IEEE Conference on Security and Privacy, Oakland, 1996.
5. M. Blaze, J. Feigenbaum, and J. Lacy, “KeyNote:Trust management for public-key infrastructure”, LNCS 1550, pp.59-63, 1999.
6. KeyNote web page, “The KeyNote Trust-Management System”, http://www.cis.upenn.edu/ keynote/.
7. Y. H. Chu, J. Feigenbaum, B. LaMacchia, P. Resnick and M. Strauss, “REFEREE:Trust Management for Web Applications”, AT&T Research Labs, 1997, http://www.research.att.com/ mstraus/pubs/referee.html.
8. A. Kini and J. Choobineh, “Trust in electronic commerce:defintion and theoretical considerations”, 31st Annal Hawaii International Conference of System Sciences, Hawaii, 1998.
9. D. W. Manchala, “Trust metrics, models and protocols for electronic commerce transactions”, Proceedings of 18th International Conference on Distributed Computing Systems, 1998.
10. G. Della-Libera et al, “Web Services Trust Language (WS-Trust)”, http://www-106.ibm.com/developerworks/library/ws-trust/, December 18th, 2002.
11. T. Grandison and M. Sloman, “A survey of trust in Internet application”, IEEE Communications Surveys, Fourth Quarter, 2000.