F. Moyano, C. Fernandez-Gago, I. Agudo, and J. Lopez, "A Task Ordering Approach for Automatic Trust Establishment", *Proceedings of the 2012 International Symposium on Engineering Secure Software and Systems (ESSoS 2012)*, LNCS vol. 7159, pp. 7689, 2012.
http://doi.org/10.1007/978-3-642-28166-2_8
NICS Lab. Publications: https://www.nics.uma.es/publications

A Task Ordering Approach for Automatic Trust Establishment *

Francisco Moyano, Isaac Agudo, Carmen Fernandez-Gago, Javier Lopez
NICS
www.nics.uma.es
University of Malaga,
Spain
{moyano,isaac,mcgago,jlm}@lcc.uma.es

Abstract

Trust has become essential in computer science as a way of assisting the process of decision-making, such as access control. In any system, several tasks may be performed, and each of these tasks might pose different associated trust values between the entities of the system. For instance, in a file system, reading and overwriting a file are two tasks that pose different trust values between the users who can carry out them. In this paper, we propose a model for automatically establishing trust relationships between entities considering an established order among tasks.

1 Introduction

Trust has become an issue of paramount importance when considering systems security. Despite of its importance, a clear, standard definition of trust has not been provided yet. However, it is wide accepted that trust might assist decision-making processes, such as those involved in authorization schemes.

If establishing a definition of trust is very important, how to measure it is also a matter of research and can vary depending on the context where it is applied and the problem that the trust model is meant to solve. What it is mostly common among all the definitions of trust is that it involves a trustor (entity that trusts) and a trustee (the entity on which the trustor places its trust). Thus, the trustor places some trust on the trustee to perform a given task. A usual example to understand this can be given by the fact that one might trust his mechanic for repairing his car but not for fixing his teeth.

*The research leading to these results have received funding from the European Community's Seventh Framework Programme FP7/2007-2013 as part of the Network of Excellence NESSoS (www.nessos-project.eu) under grant agreement number 256980. The first author is funded by the Spanish Ministry of Education through the National F.P.U. Program.

(a) Trust Graph for Task *Read a File*

(b) Trust Graph for Task *Write a File*

Figure 1: Trust Graphs for a File System with three Users

This example shows very unrelated tasks but in some cases entities of a system perform tasks that sometimes are overlapping or related. In this paper, we address the issue of how to derive trust values for entities using an established order between tasks. We consider that the trust relationships among entities in a system can be expressed as a trust graph where the edges are the trust values between two of them for a given task. Considering an order on the tasks will allow us to determine the trust graph of one task based on the trust graph of the other task.

Consider, for example, a file system with several users each of them able to perform two different tasks on it: *read a file* and *write a file*. Users of the system might trust other users to perform these tasks. The organization that owns this file system estimates that, due to privacy regulations, reading a file poses a high risk, and decides that reading a file should be more restrictive than writing a file. Two questions arise: on the one hand, how and under which criteria to define an order between these tasks? On the other hand, let us assume that the organization assigns trust values to the users of the systems for the task *read a file*. Suppose that user A trusts user B to read a file with trust value 0.3, while the same user A trusts user C with value 0.6 for the same task, as shown in Figure 1a. Is it possible to automatically derive trust values for these users regarding the task *write a file* (Figure 1b), if we know the relationship between this task and the task *read a file*? We intend to address these two questions by first defining an order on the tasks of a system and then defining a trust model that can be used for establishing the unkown trust relationships.

The paper is organized as follows. Section 2 gives an overview of similar work carried out prior to this paper. Then, in Section 3, a graph-based trust evaluation model is introduced, as it encompasses the foundations required for the rest of the paper. The order between tasks is explained in Section 4, and in Section 5 our proposed model is presented. In Section 6, our model is applied to an e-Health scenario. Finally, Section 7 presents the conclusions, as well as some relevant lines for future research.

2 Related Work

To the best of our knowledge, the idea of exploiting an order among tasks to perform automatic trust evaluations is new. However, enough work has been previously carried out in related areas. Trust models have become very important for many computer systems and networks (see [6] for a survey). One of the most critical and important issues for these models include how to quantify trust values between the entities of the systems, i.e, trust metrics. There are many ways to define trust metrics, ranging from simple, discrete models with *trusted* and *not trusted* values, to complex models using logical formulae like BAN logic [5], vector like approaches [8], or probabilities ([9] provides a survey) and fuzzy logic [13]. Flow models such as Advogato's reputation system [10] or Appleseed [17, 18] use trust transitiveness.

In [3], a formal model is proposed to compute trust metrics between two any entities in a trust graph, using sequential operators to compute a value for a given graph path, and parallel operators to compute a final value from several graph paths. Our paper follows one of its future research proposals and builds upon some of its definitions (see Section 3).

Other approaches focus on delegation and recommendation purposes, such as [14] and [7], respectively. The former proposes a formal approach to assess the trustworthiness of potential delegatees in the context of the task to be delegated, ensuring that the choice does not cause a security breach. The latter provides an approach to take subjectivity into account when performing recommendations, in such a way that the same scalar value might mean different things to two different users.

The notions of risk and trust, as well as how they relate to each other, have also been paid attention in the literature. In [16], the authors declare that trust delegation implies risk assumptions, and propose a semirring-based trust model that takes into account the evaluation of risk and privacy for trust establishment. A mechanism that takes both trust and risk into account in order to make access control decisions is presented in [11]. Likewise, [12] considers risk and trust as two important, independent factors to strengthen interactions in e-commerce. Finally, in [4], user trust is defined as an asset. Then, by using asset-oriented risk analysis, the authors analyze which threats and vulnerabilities may cause a reduction in user trust.

3 A Graph-based Trust Metric Model

Trust can be defined as the level of confidence that an entity e_1 places on another entity e_2 for perfoming a task in a honest way. As explained in [3], trust for one task can be modelled using a weighted graph where the vertices are identified with the entities of the community and the edges correspond to trust relationships between entities regarding the task. This graph actually is a weighted digraph, since any two entities in the graph do not need to have the same level of trust in each other. Now, if we consider having n tasks instead of

just one, then we would have n weighted digraphs and as a result, a multigraph.

Next we provide some definitions based on those given in [3].

Definition 1 (Trust Domain) *A trust domain is a partially ordered set* $(TD, <, 0)$ *where every finite subset of* TD *has a minimal element in the subset and* 0 *represents the minimal element of* TD*.*

The trust domain represents the set of all possible trust values that any entity of the system might hold or might place on other entities.

Each entity in the system makes trust statements about the rest of the entities, regarding the task considered for each case. Those trust statements are defined as follows,

Definition 2 (Trust Statement) *A trust statement is an element* $(Trustor, Trustee, Task, Value)$ *in* $E \times E \times T \times TD$ *where,* E *is the set of all entities in the system;* T *is a partially ordered set representing the possible tasks, where the order established on tasks is* $\preceq$*; and* TD *is a Trust Domain.*

The set $G_x = \{(e_1, e_2, x, t) \in E \times E \times T \times TD\}$ allows building the graph for task x.

Definition 3 (Trust Evaluation) *A trust evaluation for a task* $x \in T$ *is a function* $\mathcal{F}_x : E \times E \longrightarrow TD$

This function provides, for each task x and for each pair of entities e_1 and e_2, a value t of the trust domain that e_1 places on e_2 to perform x.

4 Tasks Dependencies

An order $\preceq$ amongst tasks is mentioned in Definition 2. However, it was not specified there how to define it. The main contribution of this paper consists of exploring how this order can be used in order to calculate trust values between entities. The order between tasks imposes certain conditions on the trust values of one task regarding another task. For the sake of completeness, a possible criteria to determine the order between tasks is provided in section 4.2, but any other possible criteria could be taken into account.

4.1 Motivation

Providing a task order might make easier for a trust manager the assignment of trust values to the entities which are to perform these tasks. Although a formal definition is given later, let us for now think of x_0 and x as two tasks in T that we would like to classify w.r.t. an order. Let us assume that x_0 is lower w.r.t. this order. This means that x_0 is a reference for x in order to build its trust graph. Thus, if we consider that we can deduce the gaph of the highest tasks from the lowest tasks graphs, we could semi-automatize the process of trust values assignment. Thus, starting from the lowest task/s, we could build the

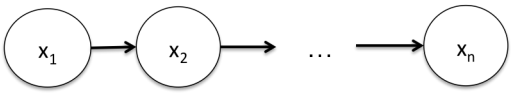


Figure 2: Task Order

graphs and trust values for the rest of tasks repeating the process downwards through the order chain, saving the trust manager the tedious work of sketching one trust graph for each task.

4.2 Task Domain

Definition 4 (Task Domain) *A Task Domain is a finite partially ordered set* $(T, \preceq)$*, where* $\preceq$ *is defined as follows: let* x_0*,* x *be two tasks in* T*.* $x_0 \preceq x$ *if and only if* $R(x_0) > R(x)$*, where* $R(x) : T \longrightarrow \mathbb{R}^+$ *is a function that given a task, returns a positive real number.*

Lemma 1 *For any given task* x*, either it is minimal or there is a minimal task* x_0 *such that* $x_0 \preceq x$

This can be easily proved if we consider the finite set $R(T) \in \mathbb{R}^+$, which represents the set of the images of the function R. Note that in $\mathbb{R}$, finite sets have a minimum element. Since T is a finite set, $R(T)$ is finite as well.

A risk assessment process, focusing on the scope and context of the task, would analyze the threats, vulnerabilities, possible losses, and other issues regarding the impact of the task on the system, and would return a number between 0 (no risk at all) and an established upper bound. Given that any risk assessment process is a rather subjective concept, we can refine this value as $R_e(x)$, which refers to the risk assessed by entity e for task x.

However, the final order between tasks should consider the opinions of all entities of the system. Thus, the final risk function (the one used for ordering the tasks) should involve the local risk functions of each entity. Let us consider an example of a system with n entities, namely e_1, e_2, ..., $e_n \in E$, and $x \in T$. $R(x)$ could be the average risk assessment, that is,

$$\frac{R_{e_1}(x) + R_{e_2}(x) + ... + R_{e_n}(x)}{n}$$

where $R_{e_i}(x)$ is the risk assessment performed by entity e_i for task x.

Once we have this value for all tasks, we can order the tasks. Let us assume that we have the order among tasks of Figure 2, where x_1, x_2, ... , x_n are the tasks in the system, and the graph declares that $x_1 \preceq x_2$ ... $x_{n-1} \preceq x_n$. Now, we can take advantage of this order to automatically build the trust graph for x_i from the trust graph of x_{i-1}, $i > 1$.

4.3 Trust Assumptions

Some assumptions about trust have been made in the definition of our model. They are listed next:

1. **Higher risky tasks should impose lower trust values among the entities**. This assumption can be reformulated as follows: if entity e_1 trusts entity e_2 with value t_1 to perform task x_1 (which is highly risky), entity e_1 could trust entity e_2 with value $t_2 \geq t_1$ to perform task x_2 (which is less risky). This also happens in real life. We would implicitly trust someone unknown higher to perform a simple, safe task than a risky, complex one. As a consequence of this assumption and our order definition, the trust values for x_i should be lower than the trust values for x_j if $x_i \preceq x_j$.

2. **Trust in entity e to perform task x is proportional to the amount of risk that entity e assigned to x**. If an entity considers that a task is risky, it will more likely take the necessary measures to ensure its successful execution.

3. **People tend to trust similar persons**. If we know that someone has a similar set of values or think in a very similar way as we do, we tend to trust that person more. In the case of entities, those entities which perform a similar risk assessment for most of the tasks will probably have similar worries and similar goals, thus they will be able to trust more each other.

4. **Mistrust should be preserved**. If entity e_1 does not trust entity e_2 to perform a task, there is not reason a priori to assume that it should trust it to perform another related task.

5 Model for Automatic Trust Values Computation

Up to now, definitions have been provided in order to establish an order on the tasks of a system. Now, we will use this order to automatically compute trust values for the entities which execute these tasks, while respecting the assumptions of Section 4.3. For this purpose, one more definition is required.

Definition 5 (Entities Divergence) *Let $e_1, e_2 \in E$ be two entities of the system. Let x_1, x_2, ..., $x_n \in T$ be the tasks of the system. We define the Entities Divergence (ED) between e_1 and e_2 as $ED(e_1, e_2) = |R_{e_1}(x_1) - R_{e_2}(x_1)| + |R_{e_1}(x_2) - R_{e_2}(x_2)| + ... + |R_{e_1}(x_n) - R_{e_2}(x_n)|$.*

Definition 5 provides a way to measure how close two entities are between them. This is the way how we incorporate assumption 3 of Section 4.3 into our model.

As we mentioned in Section 4.2, we would like to automatically generate the trust values of x_j from those trust values of x_i, being $x_i \preceq x_j$. Using the notation and definitions introduced in Section 3, what we want is to compute, for every trustor (e_1) and for every trustee (e_2), $\mathcal{F}_{x_j}(e_1, e_2)$ from $\mathcal{F}_{x_i}(e_1, e_2)$.

We have to consider all the minimal tasks as well. If we think of our model as a recursive model, in which the trust values for the current task depend on the trust values of the previous task, the trust graph for the minimal tasks would represent the base case. A graph for these minimal tasks should be sketched in order to assign the initial trust values. This assignment is made beforehand, since entities are unknown between them, and have neither knowledge nor experience with the other entities to make an informed decision on the initial trust values. However, if such information exists, it could be taken it into account during this assignment.

Note that assumption 1 in Section 4.3 states that the trust values for x_j should be higher than the trust values of x_i if $x_i \preceq x_j$. We can model it declaring that $\mathcal{F}_{x_j}(e_1, e_2)$ represents an increment over the value $\mathcal{F}_{x_i}(e_1, e_2)$. Thus, we could say that the former trust values depend on the latter ones, and at the same time, monotony property would hold: if $x_i \preceq x_j$, $\mathcal{F}_{x_i}(e_1, e_2) \leq \mathcal{F}_{x_j}(e_1, e_2)$ for any pair of entities e_1, e_2.

The question that arises is how this increment should be done, and here is where assumptions 2 and 3 of Section 4.3 come into play.

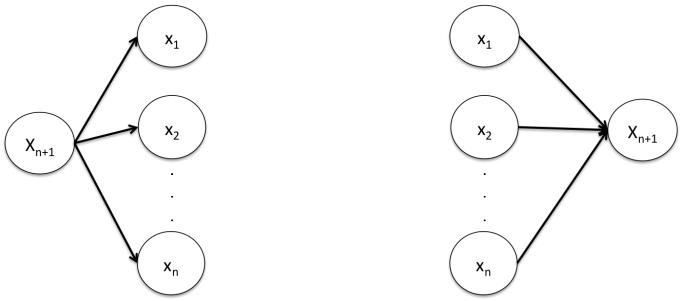
Definition 6 (Trust Incremental Value) *We define the Trust Incremental Value as a function $TIV : E \times E \times T \longrightarrow \mathbb{R}^+$ that holds the following properties:*

1. *For all entities e_1 (trustor), e_2 (trustee) $\in E$, and for all tasks $x \in T$, $TIV(e_1, e_2, x) \geq 1$*
2. *TIV should be inversely proportional to the divergence between entities, thus more similar entities will tend to trust each other.*
3. *TIV should be directly proportional to the assessed risk by the trustee. A trustee that considers a task to be very risky will be more trusted by the rest of entities for performing such task.*

As it can be noticed from the above definition, the TIV depends on both the task risk assessed by the trustee, (e_2), as stated by assumption 2, and the similarity between the trustor (e_1) and the trustee (e_2), represented by the ED (see Definition 5), as declared by assumption 3. It is out of the scope of this paper to provide a concrete definition for this value, although it would constitute an interesting future research.

Next definition explains how to compute the actual trust values:

Definition 7 (Order-dependent Trust Evaluation) *Let $x_i, x_j \in T$ be two tasks of the system, in such a way that $x_i \preceq x_j$. Let $e_1, e_2, ..., e_m \in E$ be the entities of the system. Then, for any pair of entities e_u, e_v, $\mathcal{F}_{x_j}(e_u, e_v) = TIV(e_u, e_v, x_j)\mathcal{F}_{x_i}(e_u, e_v)$*



(a) *1- n* Order (b) *n-1* Order

Figure 3: Task Dependencies Configurations

Note that assumption 4 is preserved as well, as in the case of mistrust for task x_i (i.e. $\mathcal{F}_{x_i}(e_u, e_v) = 0$), mistrust is preserved for task x_j (i.e. $\mathcal{F}_{x_j}(e_u, e_v) = 0$). Likewise, if x_i is a minimal task, the trust values for every pair of entities should be explicitly provided by the trust manager.

Also note that the purpose of our proposed trust model is to assign initial trust values. This is an important consideration, since this model does not cope with malicious entities, which might rate tasks with a high risk in order to gain higher trust. Thus, once the initial trust values have been calculated, the subsequent values are calculated with an appropriate trust model for the application, considered for each case.

5.1 Tasks Comparison

Up to now, we have only considered the case where we have a simple order configuration, as depicted in Figure 2. However, it might happen that two or more tasks are not comparable. Recall our task order definition (Definition 4). It implicitly stated that two tasks are comparable only when the total risk assessed for one of them is higher than the total risk assessed for the other one. Thus, when the risk assessment for both tasks is the same, they are non-comparable.

Consider the examples depicted in Figure 3. Figure 3a shows the case in which more than one edge comes out from task x_{n+1} (that is, n non-comparable tasks have an unique precedent task), whereas Figure 3b shows the situation in which several edges arrive in x_{n+1} (that is, x_{n+1} has many non-comparable precedent tasks). The first case does not represent any problem, since every task with an incoming arrow computes its trust values from those of x_{n+1} following Definition 7. However, the second case is more interesting, since the trust values of x_{n+1} can be computed from the trust values of tasks x_1, x_2, ..., x_n. Remember that our model is subjected to some assumptions in Section 4.3, and that assumption 1 stated the monotony property. Thus, if $x_1 \preceq x_{n+1}$, $\mathcal{F}_{x_1}(e_1, e_2) \leq \mathcal{F}_{x_{n+1}}(e_1, e_2)$ for all entities e_1, e_2. In addition, as $x_2 \preceq x_{n+1}$, then $\mathcal{F}_{x_2}(e_1, e_2) \leq \mathcal{F}_{x_{n+1}}(e_1, e_2)$, and so forth for the other tasks. Thus, if

we want to guarantee the preservation of assumption 1 we should take as the reference task that one of maximum trust value for entities e_1 and e_2.

Informally, the process would be as follows: let us assume that our system in Figure 3b has two entities, namely e_1 and e_2. We want to compute the trust value $\mathcal{F}_{x_{n+1}}(e_1, e_2)$ from the trust values of all precedent tasks x_1, x_2, ..., x_n. First, we should inspect the trust values $\mathcal{F}_{x_1}(e_1, e_2)$, $\mathcal{F}_{x_2}(e_1, e_2)$, ..., $\mathcal{F}_{x_n}(e_1, e_2)$. Then, we would choose the maximum of these values. If there are more than one maximum, we choose one of them in a indeterministic way. Assume that x_i is the task with a maximum $\mathcal{F}_{x_i}(e_1, e_2)$. Then, according to our model, $\mathcal{F}_{x_{n+1}}(e_1, e_2) = TIV(e_1, e_2, x_{n+1})\mathcal{F}_{x_i}(e_1, e_2)$. In a system with more than two entities, we would repeat this process for every pair of entities.

This is further explained in the next section, where a case study is presented.

6 Case Study: Electronic Health Records Management

In this section, we present a case study in order to provide a clearer vision on the applicability of our model. The case study has been extracted from one of the NESSoS [1] application scenarios. These scenarios are further described in [2].

6.1 e-Health

Electronic Health, or more commonly e-Health, is defined by the World Health Organization as the *use of information and communication technology for health* [15]. e-Health covers a wide range of technologies and scenarios that include interaction between patients and health service providers, as well as peer-to-peer communication and transmission of data between institutions and health professionals.

Systems that manage Personally Identifiable Information (PII) about patients require strict security measures. These systems are often referred to as Electronic Health Records (EHRs), which include patient information created by a health professional, such as laboratory reports, X-ray films, correspondence between health professionals, and so forth.

EHR repositories might be managed by different entities: General Practitioners (GPs) in their office systems, ward or hospital departments, or even by a group of hospitals that could build a circle of trust that share, under some regulations, patient information. As explained in [2], there are several scenes that arise from the EHR management problem, ranging from how to administrate policies in the parameters of EHR access control policies (e.g. groups, roles, etc), to EHR Single-Sign On and transfer of EHR data within an administrative domain. Two very interesting and frequent scenes include reading and writing into an EHR, that could be done in *emergency* mode, setting a flag that may relax the access control policies.

There are other scenarios in the context of e-Health beyond EHR management, as discussed in [2]. These scenarios might entail the use of Internet of Things (IoT) technologies to provide ubiquitous patient monitoring, and the management of patient consent for the transfer of his or her information to other administrative domains.

Since EHR management is a scoped scene, we have chosen this for the application of our model, which is presented next.

6.2 Application of the Model

We proceed to explain the model with the chosen case study.

6.2.1 Entities and Tasks.

Access control decisions might be made based on the trust level between the entities in a system. Entities might place trust on each other so that the trust system can decide whether an entity is granted access to a resource. In our example, this resource is the EHR. As explained in the preceeding section, EHR might be managed and accessed by different entities, including general practitioners, hospital departments, patients, or even groups of hospitals. For the sake of generalization, we do not make any assumptions about the type of entities, as we only consider that several entities want to access the EHR with different purposes. From now on, we refer to this entites as e_1, e_2, and e_3.

Regarding the tasks that these entities perform on the EHR, we have chosen *reading EHR*, *writing into EHR*, both in regular and *emergency* mode. We could have chosen other groups of tasks, for example, related to the EHR internal transfer scene. This would include tasks such as *transfer a record*, *transfer a group of records*, *transfer x-ray images*, *transfer hand-written scans*, and so forth. For our example, let x_1 = Reading EHR, x_2 = Writing into EHR, x_3 = Reading EHR in emergency mode, and x_4 = Writing into EHR in emergency mode.

6.2.2 Ordering the Tasks.

Up to now, we have identified the entities and the tasks of our system. Now we assume, as described in Definition 4, that a risk assessment process is carried out by each entity e_i for each task x_j by a function $R_{e_i}(x_j)$. Once each entity has assessed the risk for a task x, a final risk function $R(x)$ is applied to compute the final risk value for the task.

The left side of Figure 4 shows a possible outcome of this process. Each position in the table represents the risk assigned by the entity in that row to the task in that column. For example, the number 3 in the position (1,2) represents $R_{e_1}(x_2)$. The last row is the final risk computation for each task, that is, $R(x_i)$. We have assumed that the function R is the average risk, which is rounded down. We have also assumed that entities follow the same numeric range to assign risk values to the task (e.g. a discrete number between 0 and

ED	e_1	e_2	e_3
e_1		4	5
e_2	4		7
e_3	5	7	

Table 1: Entities Divergence Table

10). The value R (average risk in this example) determines the order $\preceq$ among

	x_1	x_2	x_3	x_4
e_1	6	3	8	7
e_2	5	5	7	7
e_3	7	4	8	4
R	6	4	7	6

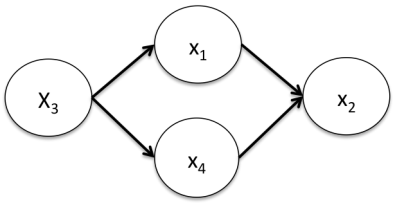


Figure 4: E-Health Tasks Dependencies

the tasks, which is depicted on the right side of Figure 4. The following relations are established: $x_3 \preceq x_1$, $x_3 \preceq x_4$, $x_1 \preceq x_2$, and $x_4 \preceq x_2$.

6.2.3 Sketching the Graph of the Lowest Task.

Since task x_3 (i.e. reading EHR in emergency mode) is the lowest task (i.e. the highest risky task), the trust manager has to sketch an initial trust graph for this task. For this purpose, the trust manager could query the entities. Let us assume that the resulting trust graph for task x_3 is the one shown in Figure 6a. Note that e_2 does not place trust on e_1, nor does e_3 on e_2. This is why there is not an arrow in these directions, although a trust relation does exist in the other way around.

6.2.4 Calculating the Entities Divergence.

The goal of the model is to compute the trust graphs for the rest of the tasks from the trust graph of the lowest task, namely x_3. The first step is to calculate how much the divergence between entities is, according to Definition 5. Considering the risk values from Figure 4, the divergence values are computed and shown in Table 1. Also note that the concept of Entities Divergence is symmetric.

Just as an example, let us compute $ED(e_1, e_3)$. According to Definition 5,

$ED(e_1, e_3) = |R_{e_1}(x_1) - R_{e_3}(x_1)| + |R_{e_1}(x_2) - R_{e_3}(x_2)| + |R_{e_1}(x_3) - R_{e_3}(x_3)| + |R_{e_1}(x_4) - R_{e_3}(x_4)| = |6 - 7| + |3 - 4| + |8 - 8| + |7 - 4| = 5.$

6.2.5 Calculating the Trust Incremental Values.

The Trust Incremental Values (see Definition 6) represent the core concept of the model. Each pair of entities have a TIV for each task. Since mistrust is

preserved in our model (see Definition 7), it is only required to compute the TIV for entities that place some level of trust on another entity in a lower task. For example, it is not necessary to compute $TIV(e_2, e_1, x_i)$ for any task x_i, as e_2 does not place trust on e_1 in the lowest task (see Figure 6a).

Our model does not impose a concrete way to compute TIV, but it just provides some criteria that should hold. According to these criteria, some possible TIVs are shown in Figure 5. These values have been established considering both the risk assessed by the trustee (represented as the column), and the entities divergence.

TIV for x_1	e_1	e_2	e_3
e_1		1.3	1.4
e_2			1.1
e_3	1.2		

TIV for x_4	e_1	e_2	e_3
e_1		1.8	1.1
e_2			1.05
e_3	1.4		

TIV for x_2	e_1	e_2	e_3
e_1		1.3	1.1
e_2			1.05
e_3	1.05		

Figure 5: TIVs for x_1, x_4, and x_2

6.2.6 Computing the Final Trust Values.

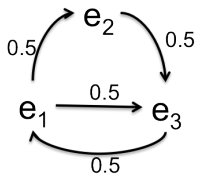
We have all the required information to apply Definition 7, that is, the actual model. At this moment, we have to remember the task order depicted in Figure 4.

We have to compute the trust values for x_1 and x_4 from the trust values of x_3 (see Figure 6a). These trust graphs are depicted in Figure 6b and Figure 6c, respectively.

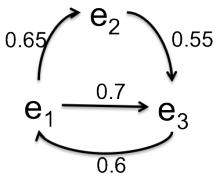
Just as an example, let us do the calculation to obtain the trust that e_3 places on e_1 to perform x_4. Applying Definition 7,

$$\mathcal{F}_{x_4}(e_3, e_1) = TIV(e_3, e_1, x_4)\mathcal{F}_{x_3}(e_3, e_1) = 1.4 * 0.5 = 0.7$$

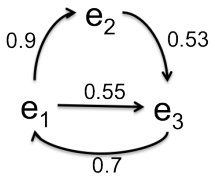
Finally, we should compute the trust values for x_2 from those of x_1 and x_4. Here we have to cope with the case described in Section 5.1, that is, to compute the trust values for one task from the trust values of more than one task. As explained in that section, if we want to ensure the preservation of assumption 1



(a) Trust Graph for Task x_3



(b) Trust Graph for Task x_1



(c) Trust Graph for Task x_4

Figure 6: Trust Graphs for EHR Scenario

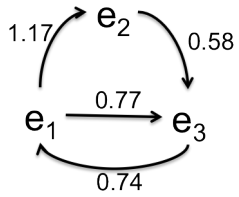


Figure 7: Trust Graph for x_2

(see Section 4.3), we have to proceed as follows: first, for a pair of entities, we examine their trust values in x_1 and x_4. Then, we chose the higher trust value, and apply the model to it, multiplying by the TIV of x_2. The final graph for task x_2 is depicted in Figure 7.

For example, to compute the trust value that e_3 places on e_1, we would chose the trust value for that entities in x_4 (0.7), thus:

$$\mathcal{F}_{x_2}(e_3, e_1) = TIV(e_3, e_1, x_2)\mathcal{F}_{x_4}(e_3, e_1) = 1.05 * 0.7 = 0.73$$

Otherwise, to compute the trust value that e_1 places on e_3, we would chose the trust value for that entities in x_1 (0.7), thus:

$$\mathcal{F}_{x_2}(e_1, e_3) = TIV(e_1, e_3, x_2)\mathcal{F}_{x_1}(e_1, e_3) = 1.1 * 0.7 = 0.77$$

After applying the model, we have the initial trust values for all the entities and tasks. From this point onward, another trust model should be in charge of updating the trust values according to the interactions between the entities. For this purpose, it might be required to map the trust values generated in our model to those trust values in the range of the latter model.

7 Conclusions and Future Work

We have proposed a model to compute trust values in a multi-task system. A multi-task system is characterized because many tasks can be performed by many entities. These tasks, in turn, impose different trust conditions between the entities. The first step in order to achieve our goal has consisted on defining a partial order between these tasks. Then, we automatize the trust evaluation process along this order while respecting some trust assumptions.

Our proposal assumes that the trust graphs for the lowest tasks are provided. Given that there is not information about the entities, this initial assignment might have an influence on the rest of the process. Further research on how to avoid or minimize the impact of this step on the whole process would be very relevant.

Furthermore, a concrete definition of TIV that respects its properties is open to future research, analyzing different alternatives and their impact on the trust values calculation.

An efficient tool implementation, through which to model the tasks order and to automatically generate and draw the trust graphs, would be interesting as future work. Finally, a validation of this tool with the NESSoS scenarios would be very relevant, as this validation could be further used to refine the model and the tool implementation.

References

[1] NESSoS FP7 Project: Network of Excellence on Engineering Secure Future Internet Software Services and Systems. http://www.nessos-project.eu/.

[2] Selection and Documentation of the Two Major Application Case Studies. NESSoS Deliverable 11.2, October 2011.

[3] Isaac Agudo, Carmen Fernandez-Gago, and Javier Lopez. A model for trust metrics analysis. In *5th International Conference on Trust, Privacy and Security in Digital Business (TrustBus'08)*, volume 5185 of *LNCS*, pages 28–37. Springer, 2008.

[4] Gyrd Braendeland and Ketil Stolen. Using risk analysis to assess user trust: A net-bank scenario. In Christian Damsgaard Jensen, Stefan Poslad, and Theodosis Dimitrakos, editors, *Trust Management, Second International Conference, iTrust 2004, Oxford, UK, March 29 - April 1, 2004, Proceedings*, volume 2995 of *Lecture Notes in Computer Science*, pages 146–160. Springer, 2004.

[5] M. Burrows, M. Abadi, and R. M. Needham. A Logic of Authentication. *ACM Trans. Comput. Syst.*, 8(1):18–36, 1990.

[6] Yolanda Gil and Donovan Artz. A survey of trust in computer science and the semantic web. *Web Semantics: Science, Services and Agents on the World Wide Web*, 5(2), 2011.

[7] Omar Hasan, Lionel Brunie, Jean-Marc Pierson, and Elisa Bertino. Elimination of Subjectivity from Trust Recommendation. In *The 3rd IFIP International Conference on Trust Management (TM 2009)*, pages 65–80, June 2009.

[8] A. Jøsang and R. Ismail. The Beta Reputation System. In *15th Bled Electronic Commerce Conference e-Reality: Constructing the e-Economy*, Bled, Slovenia, June 2002.

[9] A. Jøsang, R. Ismail, and C. Boyd. A Survey of Trust and Reputation Systems for Online Service Provision. *Decision Support Systems*, 43:618–644, 2007.

[10] R. Leiven. *Attack Resistant Trust Metrics*. PhD thesis, University of California at Berkeley, 2003.

[11] Yan Li, Huiping Sun, Zhong Chen, Jinqiang Ren, and Haining Luo. Using trust and risk in access control for grid environment. In *Proceedings of the 2008 International Conference on Security Technology*, SECTECH '08, pages 13–16, Washington, DC, USA, 2008. IEEE Computer Society.

[12] Yan Li, Ming Zhao, Huiping Sun, and Zhong Chen. A trust and risk framework to enhance reliable interaction in e-commerce. *E-Business Engineering, IEEE International Conference on*, 0:475–480, 2008.

[13] Yangjin Seo and Sangyong Han. Local scalar trust metrics with a fuzzy adjustment method. *TIIS*, 4(2):138–153, 2010.

[14] Manachai Toahchoodee, Xing Xie, and Indrakshi Ray. Towards trustworthy delegation in role-based access control model. In *Proceedings of the 12th International Conference on Information Security*, ISC '09, pages 379–394, Berlin, Heidelberg, 2009. Springer-Verlag.

[15] World Health Organization. eHealth Tools and Services: Needs of Member States, 2006.

[16] Mingwu Zhang, Bo Yang, Shenglin Zhu, and Wenzheng Zhang. Ordered semiring-based trust establish model with risk evaluating. *I. J. Network Security*, 8(2):101–106, 2009.

[17] C. N. Ziegler and G. Lausen. Spreading Activation Models for Trust Propagation. In *IEEE International Conference on e-Technology, e-Commerce, and e-Service (EEE'04)*, Taipei, March 2004.

[18] C.-N. Ziegler and G. Lausen. Propagation Models for Trust and Distrust in Social Networks. *Information Systems Frontiers*, 7(4-5):337–358, December 2005.