Trust in Global Computing

Marco Carbone[1], Mogens Nielsen[1], Vladimiro Sassone[2]

[1] BRICS*, University of Aarhus [2] Dept. of Informatics, University of Sussex

Abstract. We discuss a formal model for trust in Global Computing scenarios, focusing on the aspects of trust formation, evolution, and propagation. We focus on a particular abstract model, and illustrate its applicability on a simple example. We also discuss a possible operational model for trust based systems.

1 Introduction

In this note we briefly summarise some work on formal models of trust in Global Computing scenarios, focusing on ideas reported in [1] and [2]. The work is part of IST-FET projects SECURE and MyThS.

A GC system is composed of entities which are autonomous, decentralised, mobile, dynamically configurable, and capable of operating under partial information. For such systems, as e.g. the Internet, traditional security mechanisms have severe limitations, as they are often either too weak to safeguard against the actual risks, or so stringent to impose unacceptable burdens on the effectiveness and flexibility of the infrastructure. *Trust management systems*, whereby safety critical decision are made based on trust policies and their deployment in the presence of partial knowledge, have been proposed as an alternative in the GC setting.

The idea is basically to transfer ideas of trust as a concept in human behaviour to GC scenarios. However, it is important to realize, that there are major differences between the informal notion of trust explored in the social sciences and the kind of formality needed for GC. GC models need in the end to be operational, so as to be implementable as part of GC systems. Also, as with all formal models, they should provide a formal understanding of how trust is formed from complex interactions between individuals, and support reasoning about properties of trust-based systems.

In our approach, we think of a trust management system as consisting of a "*trust engine*" and a "*risk engine*" coupled together as part of a "*principal*." The trust engine is responsible for updating trust information based on direct and indirect observations or evidence, and to provide trust information to the risk engine as input to its procedures for handling requests. The risk engine will use the trust information in assessing the likelihood of various "*outcomes*" associated with the range of possible responses to requests, see [3] for more details. Abstracting over this point of view, we single out as central issues for our trust model the aspects of trust *formation*, *evolution*, and *propagation*. The latter is particularly important in our intended application domain, where the set of active principals is large and open-ended, and centralised trust and ad-hoc methods of propagation of its variations make little sense. An important propagation mechanism is *referencing*, whereby principals cooperate to implement complex, intertwined "global" trusting schemes.

In the following we briefly report on work towards a formal model for declarative trust policy languages with referencing [1], and some work towards an operational trust model in the form of a process calculus for trust management [2].

2 A Computational Trust Model

Principals form a set P ranged over by $a, b, c, \ldots$ and p. We assume an abstract set T of *trust values* whose elements represent degrees of trust. These can be simple values, such as {`trusted`, `distrusted`}, or also structured values, e.g. pairs where the first element represents an action, say access a file, and the second a trust level associated to that action.

As pointed out in the introduction, we model principals' mutual trust as a function which associates to each pair of principals a trust value t in T:

$$m : \mathsf{P} \longrightarrow \mathsf{P} \longrightarrow \mathsf{T}$$

Function m applied to a and then to b returns the trust value $m(a)(b) \in \mathsf{T}$ expressing a's trust in b. Note, however, that in a GC scenario a's trust values may depend on other principals' trust. For instance, a may wish to enforce that its trust in c is dependent on b's trust in c. This mechanism of relying on third-party assessments, known as *referencing*, is fundamental in all scenarios involving cooperation, including computational paradigms such as GC.

So, we refine our view of a principal's trust, assuming that principal defines its trust with a *policy*. According to such a view, each principal has a local policy π which contributes to form the global trust m. A policy expresses how the principal computes trust information given not just his own beliefs (experience), but also other principals' beliefs. It follows that a's policy π_a has the type below, whose first argument represents the knowledge of third principals' policies that a needs to evaluate π_a (in [1] we introduce a relevant example of language for describing policies).

$$\pi_a : (\mathsf{P} \longrightarrow \mathsf{P} \longrightarrow \mathsf{T}) \longrightarrow (\mathsf{P} \longrightarrow \mathsf{T})$$

By collecting together the individual policies, we obtain a function $\Pi \triangleq \lambda p.\pi_p$ whose type is (isomorphic to)

$$\Pi : (\mathsf{P} \longrightarrow \mathsf{P} \longrightarrow \mathsf{T}) \longrightarrow (\mathsf{P} \longrightarrow \mathsf{P} \longrightarrow \mathsf{T})$$

To interpret this collection of mutually recursive local policies as a global trust function m, we assume $\mathtt{T}$ to be equipped with a *complete partial order* $(T, \sqsubseteq)$ and taking Π to be continuous, we define the global trust as $m \triangleq \mathsf{lfp}(\Pi)$, the *least fixpoint* of Π.

Now the question is how to choose $\sqsubseteq$. We maintain that it *cannot* be the order which measures the degree of trust. Let T be the CPO $\{\mathtt{low} \leq \mathtt{medium} \leq \mathtt{high}\}$, and consider a policy π_a which refers to b the degree of trust to assign to c. In this setup, a will assign `low` trust to c when it is not able to gather information about c from b. This however would be an erroneous conclusion, as the interruption in the flow of information does not bear any final meaning about trust, its most likely cause being a transient network delay that will soon be resolved. The right conclusion for a to draw is not to distrust c, but to acknowledge that it does not know (yet) whether or not to trust c. In other words, if we want to model dynamic networks, we cannot allow confusion between "`don't trust`" and "`don't know`:" the latter only means lack of evidence for trust or distrust, the former implies a trust-based, possibly irreversible decision. We thus consider *approximate* trust values which embody a level of *uncertainty* as to which value we are actually presented with. Specifically, beside the usual *trust value ordering*, we equip trust values with a *trust information ordering*. While the former measures the degree of trustworthiness, the latter measures the degree of uncertainty present in our trust information, that is its information content. We will assume that the set T of (approximations of) trust values is a CPO with an ordering relation $\sqsubseteq$. Then $t \sqsubseteq t'$ means that t' "refines" t, by providing more information than t about what trust value is being approximated. With this understanding the continuity of Π is a very intuitive assumption: it asserts that the better determined the information from the other principals, the better determined is value returned by the policy.

Having pointed out the need for trust structures equipped at the same time with an information and a trust ordering, we focus on the triples $(\mathsf{T}, \preceq, \sqsubseteq)$, which we call *trust structures*. Preliminary investigations of general properties of such structures can be found in [5]. Here we illustrate a particular generic way of constructing such structures from lattices of trust values, and some of its useful properties.

When defining a trust management system, it is natural to start off with a set D of trust values, or degrees. On top of that, we are likely to need ways to compare and combine elements of D so as to form, say, a degree

which comprehends a given set of trust values, or represents the trust level common to several principals. This amounts to start with a complete lattice $(D, \leq)$, where those combinators can be considered as taking lubs or glbs of sets of values. To account for uncertainty, we define an operator I to extend a lattice $(D, \leq)$ to a trust structure $(\mathsf{T}, \preceq, \sqsubseteq)$. The set T consists of the set of intervals over D which, besides containing a precise image of D – viz. the singletons – represent naturally the notion of approximation, or uncertainty about elements of D.
Given a complete lattice $(D, \leq)$ and $X, Y \subseteq D$ nonempty subsets we say that $X \preceq Y$ if and only if $\wedge X \leq \wedge Y$ and $\vee X \leq \vee Y$ [7]. Clearly, $\preceq$ is not a partial order on the subsets of D, as the antisymmetry law fails. We get a partial order by considering as usual the equivalence classes of $\sim\, = \,\preceq \cap \succeq$. It turns out that the intervals over D are a set of representatives of such classes.

Definition 1. For $(D, \leq)$ a complete lattice, the set $I(D) = \{[d_0, d_1] \mid d_0, d_1 \in D,\ d_0 \leq d_1\}$, where $[d_0, d_1] = \{d \mid d_0 \leq d \leq d_1\}$ is the interval of D determined by d_0 and d_1.

From [1] we have that the lattice structure on $(D, \leq)$ is lifted to a lattice structure $(I(D), \preceq)$ on intervals, i.e. $(I(D), \preceq)$ is a complete lattice.

At same time, we can define an ordering on intervals which reflects their information contents: as the interval $[d_0, d_1]$ expresses a value between d_0 and d_1, the width of the interval represents the uncertainty. This leads directly to the following definition.

Definition 2. For $(D, \leq)$ a complete lattice and $X, Y \in I(D)$, define $X \sqsubseteq Y$ if $Y \subseteq X$.

As for the previous ordering, we have that [1] $(I(D), \sqsubseteq)$ is a CPO.

Example 1 (Intervals in [0,1]). Let R stand for the set of reals between 0 and 1, which is a complete lattice with the usual ordering $\leq$, and let us consider the set $I(R)$ of intervals in R. It follows from the previous results that $(I(R), \preceq)$ is a complete lattice and $(I(R), \sqsubseteq)$ is a complete partial order. The trust domain so obtained is particularly interesting, as it allows us to express complex policies. In particular, it is related to the uncertainty logic [4], where an interval $[d_0, d_1]$ in $I(R)$ is seen as a pair of numbers where d_0 is called belief and $1 - d_1$ disbelief. A a formal comparison with Jøsang's logic is currently under investigation

We conclude this section by noticing that the trust structures defined above enjoy a number of nice properties, e.g. the relation $\preceq$ is continuous with respect to $\sqsubseteq$ and, conversely, relation $\sqsubseteq$ is continuous with respect to $\preceq$ [1]. Furthermore, continuity of operators on $(D, \leq)$ lifts to continuity on $(I(D), \sqsubseteq)$, and a number of useful theorems can be proven relating the trust and the information ordering, including theorems showing the correctness of certain protocols for "proof carrying requests".

3 An Operational Trust Model

In order to focus on the operational mechanisms of trust evolution and propagation in a distributed setting, in [2] we introduce a *calculus for trust management* where principals' behaviour is accounted for. The approach is in the style of process algebras. Each principal is identified by a triple $a\{\ P\ \}_\pi$, where a is the principal's name, P the behaviour which models its actions, and π its trust policy, described in a logical language such as datalog. The dynamics of the calculus consists of interactions between principals, as for instance in:

$$a\{\ b \cdot x(P) \mid P'\ \}_\pi \mid b\{\ \phi :: a\langle e\rangle\,.\,Q \mid Q'\ \}_{\pi'} \longrightarrow \quad a\{\ P\{e/x\} \mid P'\ \}_{\mathsf{upd}(\pi,a,e)} \mid b\{\ Q \mid Q'\ \}_{\pi'}$$

Such interactions are granted according to the involved principals' policies, i.e. ϕ is a predicate which must be satisfied by policy π in order for communication to happen. Any message received implies an update of the policy, done by using the update function $\mathsf{upd}()$. The overall idea here is that policy updates are informed during a's evolution in time by its history of (un)successful interactions with other principals.

Our current work on such extended framework attempts to capture the evolutionary aspects of trust in dynamic networks, together with the study of properties and related analysis techniques of systems based on trust in such networks. In [2] we define equivalence relations for comparing policies, behaviour principals and networks of principals. In an example we show how it is possible to get two kind of networks (web of trust) and associate one network the specification of the problem (done in a classical centralized way) and the other network to a possible implementation as a way of proving that the latter is correct and sound with respect to a specific property.

Conclusion

The models introduced above build on basic ideas from trust management systems and relies on domain theory to provide a semantic model for the interpretation of trust policies in trust-based security systems. This is not the end of the story, and there are still many issues to be faced in the area.

We remark that the interval construction illustrated here can be understood in abstract (categorical) terms, as done in [5]. Moreover in [8], intervals are substituted by a more pragmatical model, i.e. event structures.

The problem of implementing distributed computations of the least fix-point has been studied in [6] which provides a distributed algorithm for computing efficiently a partial global trust function, indipendetly by the set T adopted. Finally, we have also investigated ways for expressing and studying security properties of systems based on dynamic trust evolution and propagation, such as those above. Among the many approaches to checking of security properties, behavioural equivalences are particularly appealing. A valid alternative could be designing a logic for expressing trust related behavioural properties of principals.

References

1. M. Carbone, M. Nielsen, and V. Sassone. A formal model for trust in dynamic networks. In *Proc. of SEFM*, pages 54–61. IEEE Computer Society Press, 2003.
2. M. Carbone, M. Nielsen, and V. Sassone. A calculus for trust management. To appear in Proc. of FSTTCS, 2004.
3. V. Cahill et al. Using trust for secure collaboration in uncertain environment. *IEEE Pervasive Computing Journal*, 2003.
4. A. Jøsang. A logic for uncertain probabilities. *Fuzziness and Knowledge-Based Systems*, 9(3), 2001.
5. K. Krukow. On foundations for dynamic trust management. Unpublished PhD Progress Report, available at: `http://www.brics.dk/~krukow/`, 2004.
6. K. Krukow and A. Twigg. Distributed approximation of fixed-points in trust structures. Technical Report RS-04-16, BRICS, University of Århus, September 2004.
7. U. W. Kulish and W. L. Miranker. *Computer Arithmetic in Theory and Practice*. Academic Press, 1981.
8. M. Nielsen and K. Krukow. On the formal modelling of trust in reputation-based systems. In J. Karhumäki, H. Maurer, G. Paun, and G. Rozenberg, editors, *Theory Is Forever: Essays Dedicated to Arto Salomaa*, volume 3113, pages 192–204. Springer Verlag, 2004.