DYNAMICS OF TRUST EVOLUTION
Auto-configuration of Dispositional Trust Dynamics

Christian Damsgaard Jensen and Thomas Rune Korsgaard
Department of Informatics and Mathematical Modelling
Technical University of Denmark, Bld. 321, Richard Petersens Plads, Kgs. Lyngby, Denmark

Keywords: Security auto-configuration, trust evolution, trust dynamics.

Abstract: Trust management has been proposed as a convenient paradigm for security in pervasive computing. The part of a trust management system that deals with trust evolution normally requires configuration of system parameters to indicate the user's propensity to trust other users. Such configurations are not intuitive to ordinary people and significantly reduce the usability of the system. In this paper, we propose a dynamic trust evolution function that requires no initial configuration, but automatically adapts the behaviour of the system based on the user's experiences. This makes the proposed trust evolution function particularly suitable for embedding into mass produced consumer products.

1 INTRODUCTION

Ubiquitous computing (Weiser, 1991) offers a vision of the world of tomorrow in which there are computing devices embedded in everything: vehicles, household equipment, home entertainment systems, medical equipment, clothing and even in human beings. These computing devices will communicate with one another and exchange information in order to help people with tasks in a wide range of everyday contexts. Some of the devices will be powerful computers, while others are tiny chips without much computing power, but suitable for embedding in clothing or within the organs of the human body. Regardless of their capabilities, the devices are expected to be connected at all times and in all locations, and to interact across traditional organisational boundaries. Common for many of these devices is that they will be owned, managed and operated by ordinary people who are generally unable to understand the security implications of configuring and using such devices. It is therefore important to develop security abstractions that ordinary people are able to understand and manage with a minimum of effort.

Trust management has been proposed as a convenient security metaphor that most people understand and which offers the possibility of security auto-configuration (Seigneur et al., 2003b). Existing work on trust management (Blaze et al., 1999; Grandison and Sloman, 2000; Cahill et al., 2003) focuses on making specific, security related, decisions based on general policies that do not necessarily include explicit information about context, location, credentials or the identity of other users, e.g., entity recognition may replace entity authentication (Seigneur et al., 2003a). In the context of trust management, it is generally agreed that trust is formed by a number of interaction based factors, such as personal experience, recommendations (e.g., credentials) and the reputation of the other party, but environmental factors (*situational trust*) and individual factors (*dispositional trust*) also play an important role (McKnight and Chervany, 1996). In particular, dispositional trust captures the user's propensity to trust other people, which is an important configuration parameter in a trust based security mechanism.

Previous work on trust evolution (Jonker and Treur, 1999; Jonker et al., 2004; Gray et al., 2006) operates with the notion of *trust dynamics*, which determine how trust values change over time and how trust decays when it is not actively maintained.[1] Six different types of trust dynamics have been identified, but the trust update functions that have been proposed to implement them are all statically defined. This means that the trust update function must be configured by the users before a device is used for the first time and there is no way to account for changes in people's dispositional trust based on their experience, e.g., to reflect that people who are frequently cheated are likely to become less trusting.

[1] This decay is often referred to as *ageing* or *discounting* of old experiences.

Damsgaard Jensen C. and Rune Korsgaard T. (2008).
DYNAMICS OF TRUST EVOLUTION - Auto-configuration of Dispositional Trust Dynamics.
In *Proceedings of the International Conference on Security and Cryptography*, pages 509-517
DOI: 10.5220/0001921305090517

In this paper we propose a dynamic trust evolution model, which captures the way a person's inclination to trust another person depends on the experience gained through interactions with that other person. This means that consistent good experiences will not only result in a high level of trust at a given time, but will also result in a more optimistic outlook toward the other person, i.e., that trust will grow more rapidly in the future compared to other people with whom the user may have more mixed experiences. We focus this work on the dynamics of trust evolution, which means that we do not distinguish between the different forms of interaction based factors. Recommendations and reputations may be considered equivalent to personal experience once the trust in the recommender or the reputation system has been taken into account. One of the defining properties of the proposed dynamic trust evolution model is that it requires no initial configuration of the individual trust factors, which means that it is well suited for implementation in software, which is loaded into small embedded devices developed for a mass market.

The rest of this paper is organised as follows. Section 2 defines the trust model and defines a simple graphical representation to show the relationship between experience and trust. The underlying trust evolution model is presented in Section 3 and the dynamic aspects of the trust evolution model are defined in Section 4. A preliminary evaluation of the proposed model is presented in Section 5 and our conclusions and some directions for future work are presented in Section 6.

2 TRUST MODEL

The trust model defines how trust is represented in the system in the form of *trust values* and determines the types of operations that can be performed on these trust values. These operations primarily relate to the way trust values are updated to reflect the evolution of trust in other parties.

In this paper we follow the definition of trust values made by Stephen Marsh, who defines trust as a variable in the open interval $]-1;1[$, where -1 is complete distrust and 1 is complete trust (Marsh, 1994).

The model distinguishes between entities that have been encountered before and for whom personal experience has been recorded and strangers for whom an initial trust value must be based on some system defaults. Entities about whom experience has been recorded are entered into a *Ring of Trust* which stores their trust value and the parameters that determine the way trust in each entity evolves.

2.1 Trust Evolution

In order to facilitate our discussion of trust evolution, we introduce a two-dimensional coordinate system from -1 to 1 on both the X-axis and the Y-axis (cf. Figure 1).

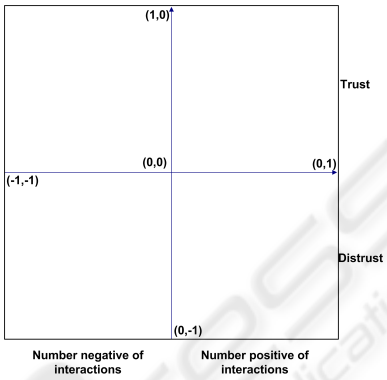

Figure 1: Coordinate system used to represent trust.

Trust is represented as a function, where the trust value is represented on the Y-axis, determining whether the trustee is in trust or distrust.[2] Each trustee has her own function in this coordinate system. The X-axis is determined by the relationship between positive and negative experiences. Calculating an X value from the previous interactions, gives the possibility of calculating a trust value as function of the X value.

These definitions result in a trust function, which will be present in the 1st quadrant and in the 3rd quadrant. The 4th quadrant is not considered, because it does not make sense to distrust an entity if there is a majority of positive experience, which would be the case if the function was placed in the 4th quadrant. Likewise, a situation where the function is present in the 2nd quadrant corresponds to a scenario where the user has trust and a majority negative interactions. The actual trust evolution function is defined in Section 4.1.

2.2 Ring of Trust

Each trustee that the trustor interacts with should have a different trust profile, and therefore a different curve, because of individual and subjective opinions about their interactions. Each curve is regarded as an instance of the trust model that represents each trustee. In order to keep track of these curves we have a trust management system. The core of this trust management system defines a *Ring of Trust (RoT)*,

[2]We refer to this relationship between experience and trust value as the trust evaluation function or sometimes simply the curve.

which is the database that keeps track of all the information that is extracted from encounters with other users in a pervasive computing environment. The information stored in the RoT consists primarily of the number of positive and negative experiences and the parameters needed to construct the curve. This RoT is consulted in all operations relating to trust initialisation and trust updates.

3 TRUST EVOLUTION MODEL

The trust evolution model has three main parts:

- Initial trust. Defines how a new trustee in the Ring of Trust is initialised.
- Trust dynamics. Defines the speed that a trustee in the Ring of Trust progresses in trust.
- Trust evolution model. Defines the actual function used for trust evolution, which also provides the basis for the dynamic trust evolution model (cf. Section 4).

These first two parts are inspired by Jonker and Treur's model of trust dynamics (Jonker and Treur, 1999), but the dynamic trust evolution model is a novel extension of the more statically defined trust update functions defined in their work. The three parts of our trust evolution model are described further in the following.

3.1 Initial Trust

Interacting with other entities in a pervasive computing environment opens the possibility of the newly encountered entity to become trusted. When a new entity is introduced into the system, it has to have some sort of initial trust value. Jonker and Treur identify two types of initial trust: initially trusting and initially distrusting, where both these approaches require a configuration of trust.

The type of trust evolution function to use depends on the user's dispositional trust and would normally have to be configured into all her devices when they are initially deployed. The configuration would be used to determine what kind of a person a user is:

- Does the user want to make quick progress or is the user more cautious? A cautious user needs a lot more positive interactions before she trusts another user, than an optimistic user needs.
- Does it take long to develop trust in another person or does the user only need one or two positive results?
- When a trusted person suddenly acts different than expected, will this destroy the relation ship or is the user more forgiving and needs several betrayals in order to loose trust?

With a system that requires an initial trust configuration, upon installation or initiation the user would have to decide on the questions above. This would determine what kind of person the user is in terms of trust and help define her trust evolution function. People, however, change their opinion over time to become more optimistic or cautious; often without being consciously aware that they changed their mind. Another point is that this configuration might not be the same for all the people in the user's ring of trusted persons.

Our goal is to have a system, which needs no configuration upon initialisation, but allows a user to develop different trust dynamics toward different entities depending on their interaction history. We therefore define a separate trust model for each person that the user is interacting with, hence the individual curves in the RoT. In order to avoid configuration the system has a neutral view of new users. Therefore all new users are given the initial trust value 0. And since there is no interactions with this new user, the user starts the trust function in point (0,0).

3.2 Trust Dynamics

Trust Dynamics describe the way positive or negative experiences influence trust. Every time a user interacts with another entity, the user is asked to tell whether the experience was positive or negative. This feedback can be used to determine the kind of relationship between the trustee and the trustor. With this feedback the trust function is also updated (cf. Section 3.3). The following six types of trust dynamics have been identified:

Blindly positive defines a trust profile, where a trustor trusts a trustee blindly after a set of positive experiences. After this set of experiences the trustee is trusted blindly for all future interactions, no matter what.

Blindly negative defines the opposite of blindly positive. After a number of negative experiences the trustor will never trust the trustee again, and the trustee will have unconditional distrust, no matter what.

Slow positive, fast negative dynamics, defines a trustor that requires a lot of positive experiences to build trust in a trustee, but it only takes a few negative experiences to spoil the build up trust.

Balanced slow defines a trustor that progresses slowly on building trust and slowly on loosing trust.

Balanced fast defines a trustor that progresses fast on building trust and looses it fast as well.

Fast positive, slow negative dynamics define a trustor that takes a few positive experiences to build trust to a trustee but takes a lot of negative experience to spoil it again.

These six approaches describe how the trust value on the Y-axis varies as a function of the recorded experience on the X-axis. One way to implement this, would be to use different granularities on the X-axis, depending on the trust dynamics, so if the user is defined as fast positive and slow negative, the granularity of the X-axis will be coarse, i.e., there will only be a few steps from 0 to 1, when adding positive experience, but fine when subtracting negative experience from the X-value.

However, choosing this approach to trust dynamics would be difficult to implement, due to the requirement that the implemented system has to be able to work without configuration. It is difficult to determine through the users actions which of the six approaches the trustor would use toward a trustee.

We therefore propose an approach, where experience influences the trust dynamics by changing the shape of the trust evolution function, thus making it dynamic.

3.3 Trust Evolution Function

Jonker and Treur define 16 required properties for a trust evolution function, but only 10 of these are relevant for developing the trust evolution function for our trust evolution model; the last 6 properties are not considered, because they deal with approximation of the trust evolution function. As we shall see later (cf. Section 5.1), the dynamic trust evolution function defined by our model does not originate from a data set but from a mathematical formula, so there is no need for approximation.

When designing a system that has no configuration, it is impossible to tell what kind of person is using the system. Therefore the system takes a neutral point of view when a new user is introduced in the Ring of Trust. The first curve is a linear function, f(x) = x, which corresponds to a balanced trust dynamics that is neither slow nor fast.

Every time the user has an experience with another entity, the user is asked to indicate whether it was a positive or a negative experience. This feedback is used to update the trust evolution function, so that it may transparently adapt to the kind of trust profile the user has with respect to the other party.

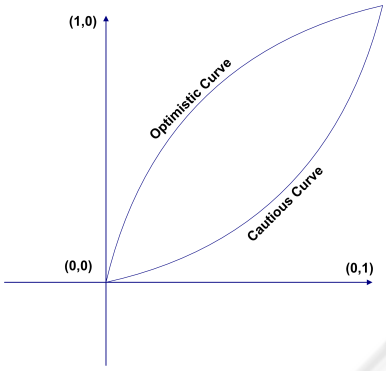

Figure 2: Trust evolution function, the cautious and the optimistic curve.

In general, people can be more optimistic or cautious when it comes to evolving trust. A user is optimistic if the user tends to trust a person based on a few experiences. If a person is more cautious it takes more interactions to obtain a high trust value, where optimistic persons get a high trust value fast. The optimistic and cautions curve is shown in Figure 2.

We see that both curves ends in trust value 1, but the optimistic curve will have higher trust values while approaching 1. Basically this means that an optimist calculates a higher trust value based on fewer interactions. On the other hand, this leads to trust that is based on less experience and therefore more easily misplaced.

4 DYNAMIC TRUST EVOLUTION

As previously mentioned, the trust evolution function must be able to change over time based on feedback from the user. It is sometimes possible to determine this feedback implicitly, e.g., a music recommendation system may recommend a song to the user, but the recommendation system may infer that this was a poor recommendation if the user asks for another song after a few bars. In many cases, however, the user will have to provide this feedback explicitly. The feedback provides an idea of how risk willing the user is.

Basically the system must deal with six different situations when getting feedback from the user, and altering the user trust values in the RoT:

1. A user can be in the trust region (1st quadrant) and have a positive experience.
2. A user can be in the trust region (1st quadrant) and have a negative experience.

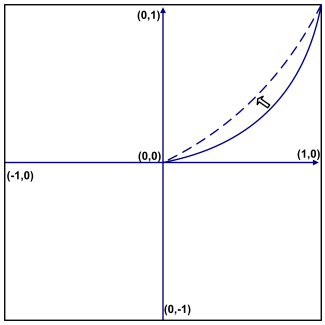

Figure 3: A user with a cautious curve has a positive experience, and the curve is updated accordingly.

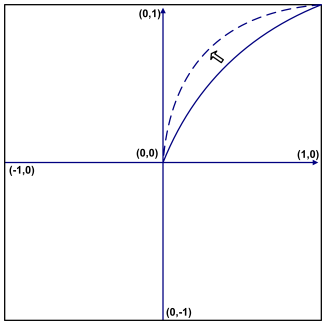

Figure 4: A user with a optimistic curve has a positive experience, and the curve is updated accordingly.

3. A user can be in the distrust region (3rd quadrant) and have a positive experience.
4. A user can be in the distrust region (3rd quadrant) and have a negative experience.
5. The user can be in Origin (0, 0), and have a positive experience.
6. The user can be in Origin (0, 0), and have a negative experience.

In the scenario where the user is in the trust region and has a positive experience, this should lead to an improvement in trust for the entities that provided the service. For the optimistic and the cautious curve the development in the evolution curve would look like Figure 3 and Figure 4 respectively.

The evolution function extends from the full-line curve to the dashed curve. We see that there is a difference in the progress of trust, depending on whether the user is an optimistic person or a cautious person.

On the other hand if the experience is negative then the trust value must be decreased for those entities that provided the service which was unsatisfactory; this is shown in Figure 5 and Figure 6.

The function moves from the full-line curve to the dashed, and again there is a different progression in

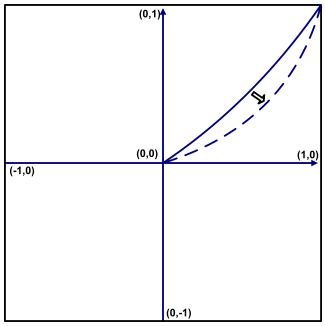

Figure 5: A user with a cautious curve has a negative experience, and the curve is updated accordingly.

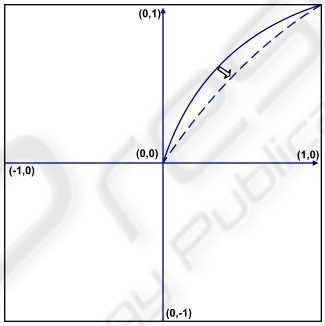

Figure 6: A user with a optimistic curve has a negative experience, and the curve is updated accordingly.

trust depending on whether the user is cautious or optimistic.

In this model it is only possible to have a cautious curve when a negative experience has been recorded and only possible to have an optimistic curve if there has been a positive interaction, which corresponds well to human intuition.

This approach for changing the trust evolution function does also apply to the distrust region. If the user has a negative experience, then the trust evolution curve is pushed further down toward (0, -1) and if the user encounters a positive experience, then the curve is pushed further up against (-1, 0).

We therefore need to define a dynamic trust evolution function that captures the behaviour that we described above. One such function is presented in the following.

4.1 Dynamic Trust Evolution Function

As described in Section 3.3 the initial trust function is defined as the function f(x) = x, which gives a neutral trust dynamics upon initialisation of a user in the RoT.

Using a simple linear formula, however does not meet the requirements that we defined above, because we are unable to represent the ways that a user's pro-

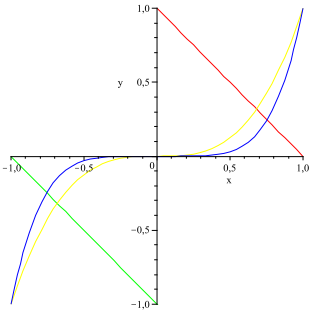

Figure 7: Trust Evolution Function represented with a polynomial expression.

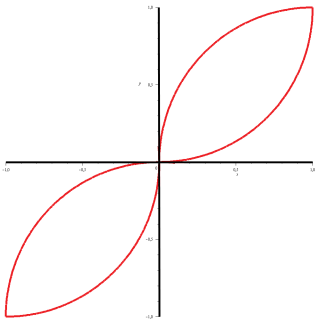

Figure 8: The trust function plotted where $a = 1$, $b = 1$ and $n = 2$.

file is changed into a more optimistic or cautious curve with a linear expression. Another approach would be to represent the function as a polynomial function with different degrees power. The disadvantage of using a polynomial function is that it is not mirrored in $f(x) = -x + 1$ and $f(x) = -x - 1$, as shown in Figure 7. By having a function expression that is not weighted equally, the steps closest to 0 on the X axis are much less significant than the steps closest to 1 and -1, which would not be fair as step toward trust should only depend on the curves parameters and not the functional expression of the curve.

A third approach (and the approach chosen) is to represent the trust function as a superellipse (or Lamé curve). The superellipse is represented by the formula:

$$\left|\frac{x}{a}\right|^n + \left|\frac{y}{b}\right|^n = 1$$

The superellipse can be adjusted by tweaking the parameters a, b and n. The parameter a and b represent the radius of the superellipse, and if they are set to 1 then the radius is 1, which fits the function with in the interval defined in Section 2.1. In this case it is only necessary to operate in the interval -1 to 1, and therefore is not necessary to adjust the a and b parameters. Hence we only need to store and adjust the parameter n to manage the shape of the dynamic trust evolution function.

However, the superellipse needs some adjustments in order to fit the desired curves as shown on Figure 3–Figure 6. Four sets of functions are defined in order to fit the different curves needed:

- Optimistic curve in trust. See Equation 1.
- Cautious curve in trust. See Equation 2.
- Optimistic curve in distrust. See Equation 3.
- Cautious curve in distrust. See Equation 4.

The four different curves are defined like this, where the parameter a and b are set to 1

$$|x-1|^n + |y|^n = 1 \quad (1)$$

$$|x|^n + |y-1|^n = 1 \quad (2)$$

$$|x|^n + |y+1|^n = 1 \quad (3)$$

$$|x+1|^n + |y|^n = 1 \quad (4)$$

These equations can be solved and the resulting contributions to the trust evolution function are shown in Table 1. The curves corresponding to these equations are plotted in Figure 8, where n is 2:

This implementation has the advantage that the neutral trust function, that has been assumed to be linear, is automatically represented for n=1.

4.2 Trust Dynamics

In our approach to trust dynamics we have made some assumptions based on the experimental research by Jonker, Treur, Theeuwes and Schalken (Jonker et al., 2004). From their research, we see that after 5 successive positive interactions and no previous interactions, the trust value is almost at a maximum. Therefore we make the definition that after 10 successive positive interactions the trust value should be at a maximum.

Based on the coordinate system where the x value goes from -1 to 1, we define a positive experience to an increase to be $\frac{1}{10}$ of the possible interval. As the experiment is based on no previous actions this interval would be from 0 to 1, hence a positive experience should increase the x value by 0.1.

The same experimental research shows that negative interactions reach almost maximum distrust on 5 recommendations as well, so we define a negative interaction should decrease the x value by 0.1.

The dynamic trust evaluation function has to distinguish between 5 different scenarios: 1) neutral user[3]; 2) optimistic user in trust; 3) cautious user in trust; 4) optimistic user in distrust; 5) cautious user in distrust. The corresponding trust update functions are shown in Table 1.

[3]In the first interaction all users are considered neutral, where $x = 0$ and $n = 1$.

Table 1: Trust values for the different scenarios.

Scenario	T(t)
1	$T(t) = 0$
2	$T(t) = (-\|(x-1)^n\|+1)^{\frac{1}{n}}$
3	$T(t) = -(-\|x^n\|+1)^{\frac{1}{n}}+1$
4	$T(t) = (-\|x^n\|+1)^{\frac{1}{n}}-1$
5	$T(t) = (-\|(x+1)^n\|+1)^{\frac{1}{n}}$

The x value and n value are updated, so that $x = x+0.1$ and $n = n+0.1$ if the experience was positive and $x = x-0.1$ and $n = n-0.1$ if it was negative.[4]

The model presented in this paper does not take temporal aspects into account. This is, however, easy to include in the model by registering when interactions take place and adjust the experience from the interactions according to their age (this effectively implements the forgetability proposed by Jonker et al. (Jonker et al., 2004)). The following example illustrates how this could be done.

We define that after 3 months an interaction is only worth 50% of its original value and after 9 months it is only worth 25% and after a year it is not worth anything any more. If a person has 2 positive experiences within the last week, a negative experience that is 4 months old, a negative experience that is 10 months old and a positive experience that is 2 years old the X value will be calculated as following:

$$XValue = 0.1+0.1-0.1\cdot 50\%-0.1\cdot 25\%+0.1\cdot 0\% = \underline{0.125}$$

5 EVALUATION

The proposed dynamic trust evolution model has been evaluated with respect to the required properties defined by Jonker and Treur and a prototype has been implemented in a recommendation system for an online service.

5.1 Properties of the Trust Evolution Function

First of all, we wish to determine whether our trust evolution function satisfies the 10 required properties, identified by Jonker and Treur.

Future Endependence. Our definition of the dynamic trust evolution function only depends on previous interactions as defined by the trust dynamics.

Monotonicity. The dynamic trust evolution function is monotonic, because the functions make sure that a higher X value can never give a lower trust value than the actual trust value.

Indistinguishable Past. We cannot determine the trust evolution curve (n value) or the trust dynamics (X value) by analysing the trust value

Maximal Initial Trust. There is a maximum trust value of 1.

Minimal Initial Trust. There is a minimum trust value of -1.

Positive Trust Extension. The proposed trust update function is monotone and continuous on the domain, so positive experiences will increase the trust value, thus meeting the requirements of positive trust extension.

Negative Trust Extension. The proposed trust update function is monotone and continuous on the domain, so negative experiences will reduce the trust value, thus meeting the requirements of negative trust extension.

Degree of Memory Based. The trust evolution function will forget about the past with the definition of forgetability.

Degree of Trust Dropping. It is possible to change the acceleration of trust dropping by the feedback from the user. By having a larger n value on a cautious curve trust drops faster.

Degree of Trust Gaining. It is possible to change the acceleration of trust gaining by the feedback from the user. By having a larger n value on an optimistic curve trust is gained faster.

5.2 Practical Experience

The proposed dynamic trust evolution model has been implemented in the Wikipedia Recommendation System (WRS) (Korsgaard and Jensen, 2008), which allows users of the Wikipedia to record and share their opinions on the quality of articles that they read in the Wikipedia. The WRS treats the Wikipedia as a legacy system, which means that it has been implemented without modifying the Wikipedia infrastructure or the underlying wiki engine. A brief overview of this work is presented in the following, but we refer the reader to Thomas Korsgaard's M.Sc. Thesis (Korsgaard, 2007) for a more detailed description and evaluation of the system.

[4]The adjustment of the n value is only based on experience, but it works well for the scenarios we have tested it with.

5.2.1 WRS Overview

Implementation of a recommender system on top of a legacy web-based system requires the ability to rewrite the content read from the Wikipedia servers (to insert the recommendations) and to capture and store the feedback (the recommendations) from the clients. A simple way to do this is to insert a web-proxy between the user and the Wikipedia. This architecture is shown in Figure 9, where the proxy executes on the user's own computer along with the browser.

The browser must be configured to use the local web proxy (this is how users opt in), which intercepts all requests to the Wikipedia (1). The proxy retrieves the article from the Wikipedia (2) along with the recommendations that are used to calculate the reputation score for the article. The article is rewritten to include the reputation score and forwarded to the browser (3). The user now has an indication of the quality of the article and may decide to provide feedback regarding the quality of the page and the utility of the reputation score (4). The user's indication of the utility of the score is used to determine whether the recommendation provided a positive or negative experience and the user's own rating is stored in the feedback repository in the Wikipedia (5). The different components are described in greater details in the following.

5.2.2 Recommendation Repository

Treating the Wikipedia as a legacy system means that there is no access to the Wikipedia's underlying SQL database, so recommendations have to be stored somewhere else. There are two obvious solutions to this problem, either to develop a separate decentralized database infrastructure or to store the recommendations as HTML comments in the pages of the Wikipedia.

A scheme with decentralised databases, where each user keeps a private database of ratings given to articles, introduces a series of problems. First of all, the amount of data kept may grow very large and the database can become too big for private users to maintain. Secondly, it introduces a problem with exchanging ratings between users, which is made more difficult by the fact that not all users are online all the time, so some ratings are not available all the time. Finally, decentralised databases open up for potential security flaws, because users must provide external access to a database on their personal computer.

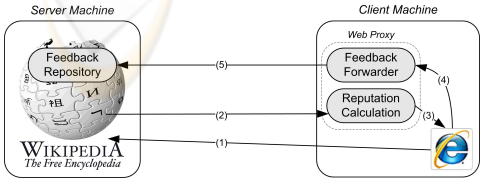

Figure 9: Overview of the Wikipedia Recommender System.

Storing recommendations directly in Wikipedia pages benefits from the fundamental Wiki philosophy of providing a central repository of information that all users can easily modify. Moreover, storing recommendations in HTML comments means that they will not be shown by existing web browsers, so the WRS is invisible to users who do not participate in the system. This also means that comments are always online and recommendations are only stored in one place. The downside to this approach is that recommendations have to be protected against fabrication and modification attacks, but this may be done by standard cryptographic techniques for authenticity and authentication as described elsewhere (Korsgaard and Jensen, 2008). Storing recommendations in the Wikipedia also leaves them open to vandalism, but it is easy to revert the existing article to an earlier version of the article if the page is vandalised. When a Wikipedia page is vandalised, the page normally restores quite fast (eBlogger, 2005).

5.2.3 Reputation Calculation

The calculation of a reputation value for an article is based on the recommendation repository, which stores all the ratings that other users have given the article. Each recommendation consists of five elements:

The mark The rating that the user has given the article.

The user The registered Wikipedia user name of the user who gave the mark. The name is chosen by the user when registering with the Wikipedia, but there may be no link to the user's real identity.

The version The version number of the article that the recommendation relates to.

The article name The name of the article is inserted into the recommendation, in order to prevent that recommendations are copied to other articles.

The hash The hash protects the integrity of the the user name, the mark and the version. The title of the Wikipedia article, the mark, and the version are concatenated and signed with self-signed certificate, where the public key is kept at the user's personal user page. This prevents ratings from being tampered with, moved to other pages or moved to a later version of an article.

The first three elements are used to calculate the reputation, while the two last elements are included to ensure the integrity and authenticity of the recommendations. The calculation of the reputation score is based on the dynamic trust evolution model defined in this paper.

5.2.4 Summary

The dynamic trust evolution model was simple to implement and required no particular configuration by the users of the WRS. Our preliminary evaluation of the ratings of recommendations (trust values) delivered by the WRS indicate that the model behaves as expected and that the results conform to human intuitions.

6 CONCLUSIONS

In this paper, we addressed the problem of auto-configuration of a trust based security mechanism for pervasive computing. We presented a novel dynamic trust evolution function that requires no initial configuration by the user, which makes it particularly suitable for embedding into mass produced consumer products. Our evaluation shows that the proposed function satisfies all the requirements of a trust evolution function and preliminary experiments indicate results that correspond to human intuition.

Directions for future work include an extension of the proposed dynamic trust evaluation function to incorporate temporal aspects of trust, such as ageing and forgetability along the lines suggested in Section 4.2. We would also like to explore different rates of adjustments of the shape of the dynamic trust evolution function, i.e., the rate of change of the n value.

REFERENCES

Blaze, M., Feigenbaum, J., Ioannidis, J., and Keromytis, A. (1999). The role of trust management in distributed systems security. In *Secure Internet Programming*, volume 1603 of *Lecture Notes in Computer Science*, pages 185–210. Springer Verlag.

Cahill, V., Gray, E., Seigneur, J.-M., Jensen, C., Chen, Y., Shand, B., Dimmock, N., Twigg, A., Bacon, J., English, C., Wagealla, W., Terzis, S., Nixon, P., di Marzo Serugendo, G., Bryce, C., Carbone, M., Krukow, K., and Nielsen, M. (2003). Using trust for secure collaboration in uncertain environments. *IEEE Pervasive computing*, 2(3):52–61.

eBlogger (2005). EBlogger: On vandalism. http://eblogger.blogsome.com/2005/10/24/on-vandalism, accessed 10 June 2008.

Grandison, T. and Sloman, M. (2000). A survey of trust in internet application. *IEEE Communications Surveys & Tutorials*, 3(4):2–16.

Gray, E., Jensen, C., O'Connell, P., Weber, S., Seigneur, J.-M., and Chen, Y. (2006). Trust evolution policies for security in collaborative ad hoc applications. *Electronic Notes in Theoretical Computer Science*, 157(3):95–111.

Jonker, C. M., Schalken, J., Theeuwes, J., and Treur, J. (2004). Human experiments in trust dynamics. In *Proceedings of the Second International Conference on Trust Management (iTrust)*, pages 206–220, Oxford, U.K.

Jonker, C. M. and Treur, J. (1999). Formal analysis of models for the dynamics of trust based on experiences. In *Proceedings of the 9th European Workshop on Modelling Autonomous Agents in a Multi-Agent World : Multi-Agent System Engineering (MAAMAW-99)*, pages 221–231, Berlin, Germany.

Korsgaard, T. R. (2007). Improving trust in the wikipedia. Master's thesis, Department of Informatics and Mathematical Modelling, Technical University of Denmark.

Korsgaard, T. R. and Jensen, C. (2008). Reengineering the wikipedia for reputation. In *Proceedings of the 4th International Workshop on Security and Trust Management*, Trondheim, Norway.

Marsh, P. S. (1994). *Formalising Trust as a Computational Concept*. PhD thesis, Department of Mathematics and Computer Science, University of Stirling.

McKnight, D. and Chervany, N. (1996). The Meanings of Trust. Technical Report 96-04, University of Minnesota, Management Informations Systems Research Center.

Seigneur, J.-M., Farrell, S., Jensen, C., Gray, E., and Chen, Y. (2003a). End-to-end trust in pervasive computing starts with recognition. In *Proceedings of the First International Conference on Security in Pervasive Computing*, Boppard, Germany.

Seigneur, J.-M., Farrell, S., Jensen, C., Gray, E., and Chen, Y. (2003b). Towards security auto-configuration for smart appliances. In *Proceedings of the Smart Objects Conference*, Grenoble, France.

Weiser, M. (1991). The computer for the 21st century. *Scientific American Special Issue on Communications, Computers, and Networks*.