Iltaf et al. EURASIP Journal on Wireless Communications and Networking 2012, 2012:119
http://jwcn.eurasipjournals.com/content/2012/1/119

EURASIP Journal on Wireless Communications and Networking
a SpringerOpen Journal

RESEARCH Open Access

Modeling interaction using trust and recommendation in ubiquitous computing environment

Naima Iltaf[1], Abdul Ghafoor[2*] and Mukhtar Hussain[1]

Abstract

To secure computing in pervasive environment, an adaptive trust and recommendation based access control model (based on human notion of trust) is proposed. The proposed model provides support to calculate direct as well as indirect trust based on recommendations. It handles situations (by itself) both in which the requesting entity has a past experience with the service and a stranger entity requesting to access the service without any past interaction with the service. It encompasses the ability to reason human cognitive behavior and has the capability to adjust in accordance with behavioral pattern changes. X-bar control chart is used to handle malicious recommendation. The defense mechanisms incorporated by proposed model against attacks such as bad mouthing attack, oscillating behavior attack and conflicting behavior attack are also demonstrated.

1 Introduction

The technological advances over last few years have revolutionized the world of computing. Computer systems (once been deemed as isolated dedicated systems) are nowadays replaced by interactive handheld smart devices. The widespread availability of the Internet has encouraged the growth of open distributed environments. In such open environments, the desire of anywhere, anytime service access is bringing Mark Weiser's [1] vision of ubiquitous computing closer to reality. It envisions densely networked world of smart and intelligent but invisible communication and computation devices interacting with each other for resource sharing and service provisioning. Lack of fixed infrastructure in ubiquitous environment promotes computational services to become as mobile as their users.

The dynamism of the ubiquitous infrastructure means that entities (which offer services) will be confronted with requests from entities that they have never met before; mobile entities will need to obtain services within environments that are unfamiliar and possibly hostile [2]. This amplifies the severity of security tribulations as entities offering and requesting services are continuously joining or leaving. Access to collaborative resources in ubiquitous environment demands some way of authenticating an entity, as well as a way of determining the extent of access that entity may have to the shared resources. In traditional computing environments access to resources is constrained by either secure authentication mechanisms or by physical network boundaries. On the contrary, in ubiquitous environment interaction and mobility of the entities imply stringent requirement on service providers about the type and level of access that they will provide to their collaborators. Consider a scenario that shows the need of a new mechanisms for access control model in open environment.

John is walking through a shopping mall. Suddenly he remembers that he has to send an important document to his colleague through an e-mail. He does not have an Internet service on his cell phone. He wants to request Internet service from any nearby device but his cell phone is an unknown device that is not pre-configured to be trustworthy to access services offered by other devices in shopping mall. An access control model to handle inter-domain service access is therefore needed, that can allow John to access services offered by other devices of network.

To handle these issues, user access to resources should be based on trustworthiness rather than the

* Correspondence: abdulghafoor-mcs@nust.edu.pk
[2]Department of Electrical Engineering, Military College of Signals, National University of Sciences and Technology (NUST), Islamabad, Pakistan
Full list of author information is available at the end of the article

traditional techniques that statically determine the access rights of the entities. To meet the security concerns in ubiquitous resource sharing environment, the model should be able to deal with devices and environment of unknown origin and also should be adaptive to the dynamics of mobile and socially motivated computing models [3].

Trust is an elementary channel of socializing in a human world. It is a human cognitive function that spurs social interactions. Recently, many researchers are working to imitate the way human assess trust, as a channel for socializing into our new security concepts for ubiquitous computing [4-13]. There emphasizes is on the computational capabilities, with less concern for systematically designed trust infrastructure resulting in ambiguous requirements for access control in ubiquitous environment. In this article, our focus is on developing a framework for access control in ubiquitous environment based on the real world characteristics of trust. Our vision is to allow mobile users to walk into any computing environment and access required services. Users can request services in the environment using various handheld or wearable devices. Study involves asserting the trust reasoning capabilities within the system which will help it to determine the level of access that entity may or may not have to shared resources. It uses recommendations from trusted services as a mean for trust to be propagated between unknown entities. However, in the open, dynamic ubiquitous environments, numerous malicious recommenders who give unfair recommendations to maximize their own gains can also exist. Therefore, incorporating mechanisms to avoid or reduce the influence of unfair recommendations is a fundamental problem for trust models in ubiquitous environments. The framework uses X-bar control chart to filter out unfair recommendations, assuming that recommendations provided by different recommenders follow the same probability distribution. The model is also sensitive to suspicious behavior and incorporates the concept of maximum achievable trust which increases as the entity continues to behave positively and is decremented each time entity shows an alternating behavior. We believe, the proposed model is the first model that has introduced adaptive policy based management to handle strategic malicious behavior and X-bar control chart to handle malicious recommendations in order to provide an attack resistant model.

2 Related work

Security is a primary concern in any computing environment. This is particularly true for ubiquitous computing environment as it allows adhoc interaction of known and unknown autonomous entities. Trust based on human notion is applied to cope with new security concerns in ubiquitous environments. Blaze et al. [14], first proposed decentralized trust-management PolicyMaker. Their trust model was based on credential verification and secures application policies to restrict access to resources and services. However it did not use recommendations to choose a suitable reputed service. Moreover, its complex computational requirements made it not feasible for ubiquitous environment. Kagal et al. [15,16] argue that large, open systems do not scale well with centralized security solutions. They instead, propose a security solution (Centaurus) based on trust management, which involves developing a security policy and assigning credentials to entities. Centaurus depends heavily on the delegation of trust to third party. Recent research trend [4-13,17-28] is to build autonomous trust management as fundamental building block to design future security framework. SECURE project [4,5] presents a formal trust and risk framework to secure collaboration between ubiquitous computer systems. It demonstrates the aspects of the trust life cycle in three stages: trust formation, trust evolution and trust exploitation.

Shand et al. [6] proposed trust mechanism with risk assessment model for resource sharing. Model also computed recommendations using a transitive combination of values. However it suffered with issue of long chains of recommendation. He et al. [7] proposed a trust model based on cloud theory. They used expected value, entropy and hyper entropy to define a trust cloud representing trust relationship. Ya-Jun et al. [8] have presented a trust-based access control model that relies on trust negotiation to establish initial trust for authenticating strangers. It is an extension of the role based access control thus suffers with its inherent issues. Jameel et al. [9] proposed a trust model based on the vectors of trust values of different entities in ubiquitous computing. The trust computation takes into account peer reputation, confidence, and history of past interaction and time based evaluation to calculate trust value. A method for detecting a malicious recommendation is presented. The model takes into account the aggregate of events in order to compute the behavior pattern of an entity. Sequence of outcome of interaction has no effect on the evolution of trust thus ignoring the relevance of event that occurred at different times. Deno and Sun [11] proposed a Probabilistic Trust Management in Pervasive Computing that takes trust value as a probability with which a device performs satisfactory interactions with its neighbor. The problem with this model is that it cannot distinguish between getting one positive outcome out of 2 interactions and getting 100 positive outcomes out of 200 interactions because in both cases the probability is equal to 0.5 [12]. Further,

we point out that even if the trustee has no past interaction with the trustier and the outcome of the maiden interaction is also negative, the model assigns 0.33 trust value. TRULLO [13] is model proposed for assigning initial trust value using single value decomposition. It sets initial trust values based on properties of user's past experiences. Ahamed and Sharmin [17] proposed a Trust-based secure Service discovery model, for a truly pervasive environment. It is a hybrid model that makes service sharing decisions based on mutual trust. It associates a security level with each service. This allows the service manager to decide which services can be shared without explicit user input. However, this security level is not used to regulate the maximum trust value to be achieved by the user i.e., the security level does not define the maximum trust possible to be earned for a user with the given history. Komarova and Riguidel [18] proposed an Adjustable Trust Model for Access Control. All parameters of this model are defined by policy set using natural language. In this model the trust value grows/declines linearly after each interaction, which is not in accordance with human behavior. Almenarez et al. [19-22] proposed a mathematical evolutionary model also known as pervasive trust management model (PTM). PTM uses historical behavior, behavior feedback and total number of interactions for calculating new trust value. The model takes into account the aggregate of events while calculating trust. We have found that PTM does not provide protection against paradoxical behavior as claimed in [22]. An attacker can easily manipulate the system to gain high trust despite of having paradoxical behavior. Omnipresent Formal Trust Model (FTM) [23,24] presents a flexible trust model incorporating a behavioral model to handle interactions. However, it fails to handle situations where a malicious user can launch strategic attack as the trust value is not modified considering the old behavior pattern.

3 Definitions and theoretical framework

Here we present the definitions and framework for our model.

Pervasive environment: Pervasive environment is a framework or milieu, in which autonomous entities (also referred to as pervasive device) interact with each other. The interaction may be anonymous or one-way. Pervasive entities tend to trust each other for service sharing and congregate to emulate a social group.

Entity: Entity is a generic term used for a subject that accesses services provided by a service provider; it can be a user, pervasive device or requesting service itself.

Service: Each service provides some functionality and can be accessed by entities or other services. Services are provided by different service providers. A pervasive environment has set of services with their associated trust requirements and other security policies.

Service policy: Service policy defines set of rules associated with the service. An entity must conform to service policy in order to access that service.

Service interface: A service usually has different levels, each addressing different set of users. Service interface defines these levels within the same service.

Security level: Each service maintains a set of numerical values that defines the security level of its service interfaces. This security level defines the rate of reward/penalty after each interaction.

Trust: A concrete and mathematical definition of trust that has been followed in this article is given by Diego Gambetta [29]:

Trust (or, symmetrically, distrust) is a particular level of the subjective probability with which an agent assesses that another agent or group of agents will perform a particular action, both before he can monitor such action (or independently or his capacity ever be able to monitor it) and in a context in which it affects his own action.

When a service trusts some entity, it implicitly mean that the past behavior of the entity was high enough or at least not that detrimental for the service to consider providing him the access.

Trust representation: The level of trust a service can have on an entity is represented by a trust value. A higher trust value corresponds to a higher probability that an entity can be trusted. In our approach, 0 corresponds to total absence of trust. This can occur only if the service completely distrusts an entity. The table below outlines trust levels, their corresponding range of trust values and their meaning as used in our trust model.

4 Characteristics of our trust model

The proposed framework is designed for secure collaboration between known and unknown entities in an uncertain environment. The model calculates trustworthiness of each entity, analyze the behavior pattern of entity and provide service access decision in compliance with security policies. The framework has following characteristics:

- Trust is a relationship established between an entity and a service for a specific service interface thus representing the amount of trust a service has, in an entity, to authorize access to the particular service interface. Model incorporates the trust dimensions of subjectiveness, time and context. The notation {Entity *E;* Service *S;* ServiceInterface SI; Time *t*} is used to describe a trust relationship.
- The discrete levels of trust are used in this model, referred to as the trust value.

$$0 \leq T_i(E_i, S_i, \mathrm{SI}_i, t_i) < 1$$

• There are three main sources to establish trust. Direct trust is computed on the basis of experiences the entity had with the requested service (T_{dir}). When the system has no personal interactions with the entity in question, a communicated opinion about the trustworthiness of an entity can be requested $\mathrm{T}_{\mathrm{recom}}$ (indirect trust). Initially new entities joining a pervasive environment for the first time have neither evidence of past experiences nor any reference. In this case ignorance value (T_{iv}) is assigned which can be updated as additional information becomes available.

$$\exists T_i(E_i, S_i, \mathrm{SI}_i, t_i) = \{T_i(E_i, S_i, \mathrm{SI}_i, t_i) \rightarrow T_{\mathrm{dir}}(E_i, S_i, \mathrm{SI}_i, t_i) \vee T_{\mathrm{recom}}(E_i, S_i, \mathrm{SI}_i, t_i) \vee T_{\mathrm{iv}}(E_i, S_i, \mathrm{SI}_i, t_i)\}$$

• Trust is service interface specific. An entity may have different trust values for same service but different interfaces.

$$T_i(E_i, S_i, \mathrm{SI}_i, t_i) \neq T_i(E_i, S_i, \mathrm{SI}_j, t_i) \quad i \neq j$$

• Trust is a time variant value, it decays with time given that entity has no new interaction. The trust an entity has acquired at time t in a perspective of a specific service might not be the same as the trust attributed to him in the same perspective, at time $t + \Delta t$

$$T_i(E_i, S_i, \mathrm{SI}_i, t + \Delta t) < T_i(E_i, S_i, \mathrm{SI}_i, t)$$

• Social trust affects the trust factor. An entity is more likely to be trusted if it is trusted by the peer services as compared to the other services located in other autonomous pervasive environments. The trust of other services provides a basis of assigning maiden trust value.

• Trust value increases with good actions and decreases with bad actions.

$$\{(I_{\mathrm{cur}}+ \rightarrow T_i(E_i, S_i, \mathrm{SI}_i, t_i) \geq T_{i-1}(E_i, S_i, \mathrm{SI}_i, t_i))\}$$
$$\{(I_{\mathrm{cur}}- \rightarrow T_i(E_i, S_i, \mathrm{SI}_i, t_i) \leq T_{i-1}(E_i, S_i, \mathrm{SI}_i, t_i))\}$$

• Counters the suspicious behavior by limiting the maximum achievable trust value. Model also monitors entity for constant positive actions to increase maximum achievable trust.

$$\{c_{n_i} > c_{n_{i-1}} \rightarrow T_{\max_i} < T_{\max_{i-1}}\}$$
$$\{c_{p_i} > c_{p_{i-1}} \rightarrow T_{\max_i} > T_{\max_{i-1}}\}$$

• Reward/penalty rates change with the behavior of entity. Penalty factor increases with the consecutive negative behavior. Reward factor increases with the consecutive positive behavior.

$$\Delta n_i = \{\Delta n_i \,|\, c_{n_i} > c_{n_{i-1}} \wedge I_{\mathrm{cur}}- \rightarrow \Delta n_i \geq \Delta n_{i-1}\}$$
$$\Delta p_i = \{\Delta p_i \,|\, c_{p_i} > c_{p_{i-1}} \wedge I_{\mathrm{cur}}+ \rightarrow \Delta p_i \geq \Delta p_{i-1}\}$$

5 Architecture of our trust model

It is widely acknowledged that traditional security measures fail to provide necessary flexibility for interactions between known and unknown entities in an uncertain environment. This leverages us to design our trust based security architecture based on human notion of trust to allow access to resources in an uncertain environment. In our model, we assume that all entities are autonomous and some of them are mobile. Entities in our model try to access the services. Thus, we establish trust relationships between entities and the services. Each service maintains a list of trustworthy and untrustworthy entities, the trust value associated with them, and time when trust value was last revised and number of interaction the entity had with the service. An overview of our proposed trust-based security framework is shown in Figure 1. The framework consists of three main layers. The model allows service requestor to access a particular service interface or shared resource in the network on the bases of its trust value maintained in trust repository of each service. If no prior trust information is available, recommendation evaluator module seeks recommendation from peer services located within the same pervasive environment or from trusted parties offering same service located in other autonomous pervasive environments. The recommended trust value computed by indirect trust computation module form the basis for new trust relationship. Similarly, if no recommendation is available for the entity,

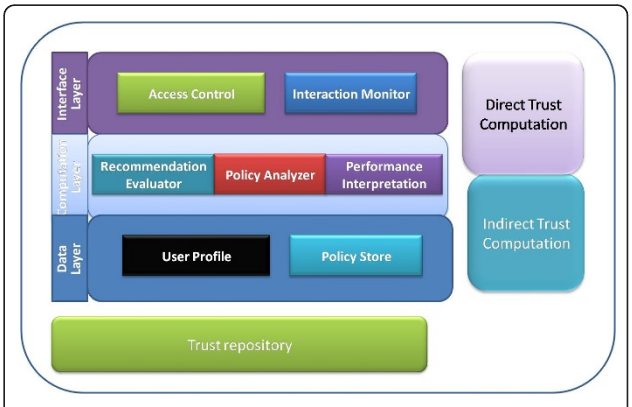

Figure 1 Proposed model.

the service can assign it an ignorance value based on the security level of service interface the entity is requesting. Performance interpretation module is responsible for the evolution process. It evaluates the behavior patterns of entity involved in interaction according to its actions as additional evidence becomes available. It is connected with Trust repository and interaction monitoring module of the system. Direct trust computation takes place after culmination of an interaction and obtaining some observation from the interaction monitoring module. The basic function of a policy analyzer is to process the request; to determine whether the requestor is permitted to do the requested action in presence of the policies defined for that service interface.

5.1 Indirect trust computation

Indirect trust computation holds key importance where requesting service has no personal interactions with the entity in question to run a direct trust computation. Indirect trust computation (Algorithm 1) is carried out by Reputation evaluator module. It seeks recommendation for further information when the amount of observation is insufficient for the service to define the trustworthiness of the entity requesting the service. It requests recommendation, with respect to the entity in question, from peer services located within the same pervasive environment or from trusted parties in other autonomous pervasive environments. The reputation evaluator module computes the recommended trust value of entity E_i for service S_i and service interface SI_i at time t as $T_{\mathrm{recom}}(E_i, S_i, \mathrm{SI}_i, t)$ based on peer services recommended trust value and other services in other autonomous pervasive environment's recommended trust value given as

$$T_{\mathrm{recome}}(E_i, S_i, \mathrm{SI}_i, t) = \alpha T_p(E_i, S_i, \mathrm{SI}_i, t) + (1-\alpha) T_o(E_i, S_i, \mathrm{SI}_i, t)$$

The weights given to peer services recommended trust value (T_p) and other services in other autonomous pervasive environment's recommended trust value (T_o) are α and (1-α) respectively where α is a positive constant that can be fine tuned to have trust value for an entity between 1 and 0. In both cases recommended trust value is computed as average of the product of the trust value and the confidence level on that trust value of all the recommenders. The peer recommended trust value is computed as average of the product of the recommended trust value and the confidence level (CL) on that trust value of all the recommenders.

$$T_p(E_i, S_i, \mathrm{SI}_i, t) = \frac{\sum_{j=1}^{N} T_j(E_i, S_j, \mathrm{SI}_k, t) * \mathrm{CL}}{N_p}$$

where N_p is total number of peer recommendation and SI_k represent any ServiceInterface. Similarly, if N_o represent total number of recommendations from other autonomous environments then recommended trust value from other autonomous pervasive environment's is given as:

$$T_o(E_i, S_i, \mathrm{SI}_i, t) = \frac{\sum_{j=1}^{N} T_j(E_i, S_j, \mathrm{SI}_k, t) * \mathrm{CL}}{N_o}$$

Confidence level: CL measures the reliability of the recommending service. It defines how certain we are about the trust value recommended by the recommending service and is given by:

$$\mathrm{CL} = \eta * \gamma * \mathrm{SL} \quad 0 \le \mathrm{CL} \le 1$$

where η is normalized interaction value, γ is Time based Experience and SL is security level of recommending Service Interface.

5.1.1 Time based experience

As much as change is about adapting to the new, it is about detaching from old. Human nature study shows that all relations show a liability of newness in which the rate of decay slows over time [6]. Trust based access control mechanism is based on the human notions and uses history of interaction for trust value computation. If each experience is given same weight in trust computation regardless of the time when the interaction happened, the computed trust value can lead to false results. Instead, the services having old experiences with the entity in question should have a less weight in peer recommendation than the new ones. Older experience should decay with time and has a less effect. Let t and t_c denote the time of last interaction and current time respectively then decay function denoted by γ is defined as:

$$\gamma(t, t_c) = \alpha(1-\beta)^{\Delta t}$$

where $\Delta t = t_c - t$, α and β are adjustable positive constant that can be tuned accordingly to define the rate of decay. Figure 2 depicts that, as the time since last interaction grows, the impact of recommended value in trust calculation decreases.

5.1.2 Effect of experience

Recommended trust is dependent on experiences of the entity with the requested service. An experience is the result of interaction with an entity. The greater the number of interactions of the recommending peer service with the entity in question, the greater is the confidence level. Hence, the confidence level on the recommender is directly proportional to the number of

interactions it had with the entity. Since trust value is given by $0 \le T_i \le 1$, we require a normalization function that can limit the number of interactions in the range 0 to 1. The function we have used to normalize interaction value (n_t) is given as:

$$\eta = \frac{n_t - n_t^{\min}}{n_t^{\max} - n_t^{\min}} \quad \text{where } 0 \le \eta \le 1$$

where $n_t^{\max}$ and $n_t^{\min}$ represent the maximum and minimum number of interaction, $n_t^{\min} = 1$ and $1 \le n_t^{\max} \le \infty$. Assuming that $n_t^{\max} = 50$, the Figure 3 depicts that recommended trust value is directly proportional to the number of interactions the recommending service had with the entity.

5.1.3 Effect of sensitivity of recommender

The recommendation based trust value calculation process, while evaluating the trust value given by the service, takes into account the sensitivity of the service interface offering the recommendation. For example, a file service has multiple interfaces depending on the type of functionality it provides and each service interface has a security level associated with it. The sensitivity of the recommender depends on service interface security level. Figure 4 shows that recommended trust value is dependent on sensitivity of the recommender (SL).

Judging the recommendation: Malicious entities often manipulate the indirect trust calculation by sending false recommendations to lower or increase the recommended trust value of the requesting entity. These malicious recommendations can highly influence the access control mechanism. In our model, we propose a simple mechanism based on control charts to determine whether a recommendation is honest or is malicious. Once the recommendation evaluator module collects all the recommendations, prior to calculating the indirect trust, these recommendations undergo a filtration process based on X-charts. X-charts consist of two limits, upper control limit and lower control limit. These two limits define a region in which recommendations falls if they are honest. It is assumed the recommendation are normally distributed with known mean μ and known standard deviation σ then upper control limit (UCL) and lower control limit (LCL) are

$$\text{UCL} = \mu + \frac{3\sigma}{\sqrt{n}} \tag{1}$$

$$\text{LCL} = \mu - \frac{3\sigma}{\sqrt{n}} \tag{2}$$

n being the size of recommendations. The steps to follow for judging the recommendations by constructing the control charts are:

(1) Collect the recommendations for the entity in question.

(2) Calculate the mean and standard deviation of the recommendation sample.

(3) Calculate the control limits.

(4) Verify if the recommendation lie within the control limits.

(5) Discard out of bound recommendations.

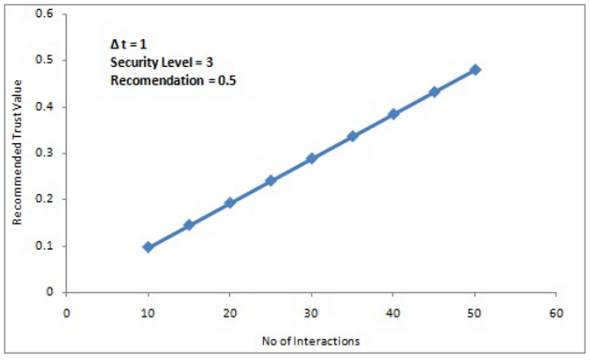

Figure 3 Effect of experience on recommendation.

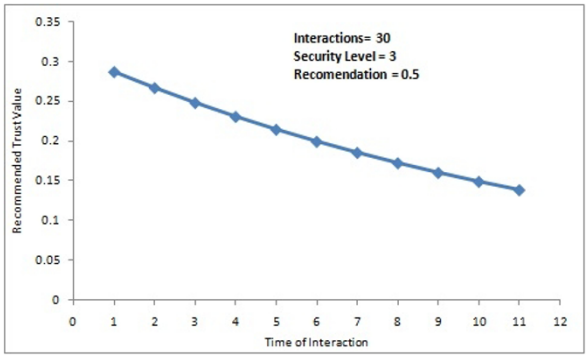

Figure 2 Effect of time on recommendation.

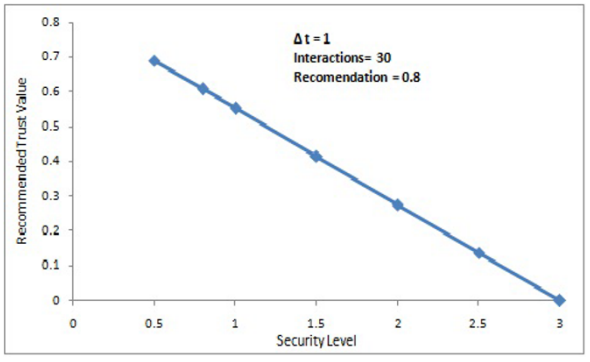

Figure 4 Effect of experience on recommendation.

5.2 Direct trust computation

Performance interpretation module is responsible for the direct trust computation (Algorithm 2). It evaluates the behavior pattern of the entity involved in an interaction according to its actions, as additional evidence becomes available. Each service maintains the following information for each entity that is updated during trust evaluation:

- No of positive interactions with the entity n_p
- No of negative interactions with the entity n_n
- No of times the entity has oscillated between positive and negative behavior $\text{OnOff}_{\text{count}}$
- No of continuous positive interactions with the entity c_p
- No of continuous negative interactions with the entity c_p
- Maximum trust that an entity can achieve on a given set of interactions $T_{\max}$
- Entity ever being blacklisted *isBlackListed*
- Entity ever being distrusted *isDistrusted*

All services residing in a pervasive environment do not need the same level of security. A weather service can be offered with even low trust value to an entity, and with frequent positive interactions entity can quickly become the trusted user of the service. But Internet services are more sensitive and require more positive behavior before declaring an entity completely trusted. Similarly negative interaction for Internet service is required to decline the trust at a higher rate as compared to the weather service. Model associates a security level (sl) value with each service to control rate of reward/penalty after each interaction. Each service maintains a numerical value that signifies the security level of the service.

Algorithm 1 *Recommendation*
Require: *Recommendations*
Ensure: *RecommendedTrust*
1: **if** E_i isStranger = true **then**
2: Broadcast Recommendation Request
3: **while** (*isReply*) **do**
4: $T_i = getRecommendation(E_i, S_i, SI_i, t_i)$
5: $i++$
6: **end while**
7: $n = i$ // n is total no of recommendations received
8: compute mean μ and standard deviation σ of all recommendation
9: compute *UCL* and *LCL* using equation 1 and 2
10: for each recommendation T_i
11: **while** $j \le n$ **do**
12: **if** $T_j > LCL$ **and** $T_j < UCL$ **then**
13:

$$n_j = \frac{n_{t_j} - n_t^{min}}{n_t^{max} - n_t^{min}}$$

$$\Delta t_j = t_c - t_j$$

$$\gamma_j = \alpha(1-\beta)^{\Delta t_j}$$

$$CL_j = n_j * \gamma_j * SL_j$$

14: **if** $S_j \in$ *Peer Service* **then**
15:

$$T_p = \frac{(T_p + (T_j * CL_j))}{n_p + 1}$$

16: **if** $S_j \in$ *AutonomousService* **then**
17:

$$T_o = \frac{(T_o + (T_j * CL_j))}{n_o + 1}$$

18: **end if**
19: **end if**
20: **end if**
21: j ++
22: **end while**

$$T_{recom} = \alpha T_p + (1-\alpha) T_o$$

23: **end if**
24: **return** T_{recom}

Trust evaluation takes place after completing an interaction. If the entity has demonstrated positive behavior during an interaction, its number of positive interactions with the entity n_p is incremented. Otherwise, the interaction is considered negative and the number of negative interactions with the entity n_n is incremented. c_p and c_n are counter number of continuous positive and negative interactions respectively. c_p is incremented when consecutive positive interactions are performed and is set to 0 when entity show change in behavior. c_n is incremented in similar way on consecutive negative interactions. Depending on the outcome of the interaction, a positive behavior is rewarded by increasing service trust in the entity and negative behavior is penalized by reducing the service trust in the entity. The updated trust value is calculated using the previous trust value and impact of current interaction in the form of reward/penalty rate using following equation:

$$T_i = \begin{cases} T_{i-1} + \Delta p & \text{for } I_{\text{cur}} = \text{positive interaction} \\ T_{i-1} - \Delta n & \text{for } I_{\text{cur}} = \text{negative interaction} \end{cases}$$

where "I_{cur}" indicates current interaction, T_i and $T_{i\text{-}1}$ indicate new trust value and previous trust value

respectively. The reward Δp and penalty rate Δn for each type of behavior is dependent on

- Total no of interactions of entity n_t
- Total no of positive interactions of entity n_p
- Total no of negative interactions of entity n_n
- Counter for consecutive negative interactions of entity c_n
- Counter for consecutive positive interactions of entity c_p
- Security level sl where $0.5 \leq sl \leq 3$.
- Slope rate σ_p for positive and σ_n for negative interaction is defined by on/off counter. Each time entity changes its behavior from positive to negative σ_n is changed by

$$\sigma_{n_i} = 2^{\sigma_{n_{i-1}}}$$

and each time entity changes its behavior from negative to positive, slope rate σ_p is changed by

$$\sigma_{p_i} = \frac{\sigma_{p_{i-1}}}{2}$$

For positive behavior reward rate Δp is calculated as:

$$\Delta p = \alpha \frac{n_p}{n_t} * 2^{\sigma_p * c_p * sl} \quad (3)$$

where α is a constant and its value is 0.01. Reward rate increases with consecutive positive interactions. Similarly for negative behavior penalty rate Δn is calculated as:

$$\Delta n = \alpha \frac{n_n}{n_t} * 2 \frac{\sigma_n * c_p}{sl} \quad (4)$$

Penalty rate Δn is dependent on the sensitivity of the relationship. According to the formula, a very trustworthy entity is not declared distrustful after just one or two bad interactions. But if negative behavior persists the entity is penalized rapidly. Penalty factor increases with the consecutive negative behavior. In general, the negative behavior converges to 0 affirming an entity as completely distrustful and positive behavior converges to 1 affirming an entity as completely trustful.

5.2.1 Time based aging of trust value

Trust is time variant; it decays with the passage of time. When an entity sends a service access request message, the specific service first checks the trust repository for the entities previous trust value to decide the level of access the requesting entity can have. The proposed model incorporates time based aging on that trust value. The trust value updated long time ago decays with the time and does not carry same weight as the one updated recently. Let t and t_c denote the time the trust value was last updated and current time respectively then the same decay function γ as defined for computing time based experience is used:

$$\gamma = \alpha(1-\beta)^{\Delta t}$$

Algorithm 2 *updateTrustV alue*
Require: *EntityE, SecurityLevels*
Ensure: *newTrustV alue*
1: **if** I_{cur} = *positive* **then**
2: **if** I_{last}! = *positive* **then**
3: $c_n = 0$

$$\sigma_{p_i} = \frac{\sigma_{p_{i-1}}}{2}$$

4: **else**
5: $c_p = c_p + 1$
6: **end if**
7: **if** $c_p \geq c_{posTh}$ **then**
8: increment T_{max}
9: **end if**
10: $n_p + +$
11: Δp = calculate increment
12: $T_i = Min\ (T_{i\text{-}1} + \Delta p,\ T_{max})$
13:**else**
14: **if** T_{last}! = *false* **then**
15: $c_p = 0$
16: $OnOff_{count} + +$
17: **if** $OnOff_{count} \geq OnOff_{countTh}$ **then**
18: Distrust (E)
19: **end if**
20: **if** $OnOff_{count} > 1$ **then**
21: Decrement T_{max}

$$\sigma_{n_i} = 2^{\sigma_{n_{i-1}}}$$

22: **end** if
23: **else**
24: $c_n + +$
25: **end if**
26: $n_n + +$
27: Δn = calculate increment
28: $T_i = Man(T_{i\text{-}1}$ - $\Delta n, 0)$
29: **if** $T_i < = 0$ **then**
30: Distrust(E)
31: **end if**
32: **end if**
33: **resturn** T_i

Where $\Delta t = t_c$ - t. α and β are adjustable positive constant that can be tuned accordingly to define the rate of decay. The impact of time based aging on trust value is calculated as:

$$T_i = T_i * \gamma \quad (5)$$

5.3 Policy analyzer

Policies provide a more constrained means for adaptive behavior of entities [30]. The model proposes an adaptive approach to handle strategic malicious behavior through a policy based management. It incorporates a set of rules for strategic attack detections, together with appropriate actions and controls to counter these attacks. The set of access policies used for trust computing are

(1) Trust value symbolizes the level of trust a service has on an entity. Different Trust levels and their corresponding Trust values are described in Table 1
(2) Rate of reward/penalty is controlled by service security level that is defined by the service.
(3) Entity is distrusted when current trust approaches to **0**.
(4) Distrusted entity is allowed to interact again after forgiveness time.
(5) $T_{\max}$ of entity is decremented and is made equal to its current trust value each time entity changes its behavior
(6) $T_{\max}$ is incremented if continuous positive behavior of an entity exceeds c_{posTh}
(7) Entity is black listed if it is distrusted **d** times, where $1 \leq d \leq 3$
(8) Entity is distrusted when $\text{OnOff}_{\text{count}}$ approaches to $\text{OnOff}_{\text{countTh}}$
Algorithm 3 illustrates the working of proposed model in the presences of policy analyzer to provide access to requesting service.

6 Malicious attacks and defense solutions in proposed model

The open nature of ubiquitous environment makes the access control models designed for this environment vulnerable to attackers. These attackers are motivated by selfish intent to either get illegitimate access to services or manipulate the reputation of others for their own benefit. In this section, we investigate attacks against trust and reputation models and how proposed model protects against these attacks in order to provide an attack resistant model.

Table 1 Trust levels and their description

Level	Value	Meaning	Description
l_0	0	Distrust	Completely untrustworthy
l_1	0 ≤ value < 0.25	High distrust	Lowest possible trust
l_2	0.25≤ value < 0.5	Low trust	Not very trust worthy
l_3	0.5≤ value < 0.75	Medium trust	Mean trustworthiness
l_4	0.75 ≤ value < 1	High trust	More trustworthy than most entities
l_5	1	Complete	Completely trust this entity

6.1 Bad mouthing attack

In bad mouthing attack, one or more entities falsely provide dishonest recommendation either to elevate trust values of malicious entities or to lessen the trust values of honest entities. The proposed model uses various mechanisms to avoid and detect this attack. Model uses X-bar control chart to filter out unfair recommendations, assuming that recommendations provided by different recommenders follow the same probability distribution. Let us take the data set of 15 recommendations :{0.2,0.3, 0.25, 0.25, 0.3, 0.8, 0.8, 0.8,0.9, 0.9, 0.7, 0.85,0.75, 0.76, 0.9} We assume that 30% of the recommenders are providing malicious recommendation ($T < 0.5$)

Algorithm 3 *permitService*
Require: *EntityE, TrustValueT*
Ensure: *AccessLevel*
1:**if** E_i isStranger = false **then**
2: **if** (*isBlacklisted*($E_{i)=true}$) **then**
3: **return** *denyAccess*
4: **else if** (*isDistrusted*($E_{i)=true}$) **then**
5: **if** ($\Delta t < t_{forgiveness}$ **then**
6: **return** *denyAccess*
7: **else**
8: T_{dir} = *searchTrustV alue*(E_i)

$\Delta t = t_c - t$

9: **end if**
10: **end if**
11: **else**
12: *requestRecommendation*()
13: **if** ($n_p + n \neq 0$) **then**
14: T_i = Compute T_{recom}
15: **else**
16: T_i = assign ignorance value
17: **end if**
18: **end if**

$l_i = mapTrustLevel(E_i, T_i)$

19: **return** grantAccess of Level l_i

to lessen the trust value of entity in question. Figure 5 shows how malicious recommendations are detected using X-bar control chart. The recommendations that lie within the interval specified by LCL and UCL are considered honest, outliers are discarded. The proposed method was able to detect malicious recommendations 100%. However, 20% valid recommendations are also discarded. Secondly, the proposed model also calculates the confidence level on the recommended trust value to

diminish the effect of fake recommendations. This mechanism does not avoid the bad mouthing attack, but it could minimize its effects. Confidence level is dependent on the size of experience, time of last interaction and also the sensitivity of recommending entity. The size of experience is a measure of the number of times the two entities have interacted. We use the size of past experience to give more relevance to the services that know the entity in question for a long time. Accordingly, we assume that the trust level of entity with more past experience has already converged to a steady trust value and therefore its judgment should be more relevant than the judgment of an entity that has less number of interactions with the entity in question. The proposed model distinguishes between old and recent interaction, giving less weight to valid but old recommendations.

Thirdly, trust propagation chain is categorized; recommendations from peer services are given more weight in recommended trust calculation process in respect with recommendation from services in other autonomous pervasive environments

6.2 Oscillating attack

Malicious entities show oscillating behavior by behaving well and badly alternatively, hoping that they will not be detected and their trust value will continue to grow while causing damage. Attackers attempts to exploit the dynamic properties of trust through inconsistent behaviors. In proposed model Performance Interpretation module attempts to judge the behavior of the entity and decreases the trust value in proportion to the number of negative actions and its on/off behavior count. The final trust value is always less than the initial trust value, as the maximum achievable trust is decremented each time an entity shows an alternating behavior. Figure 6 shows that regardless of the fact that the number of positive interactions are greater than the number of negative

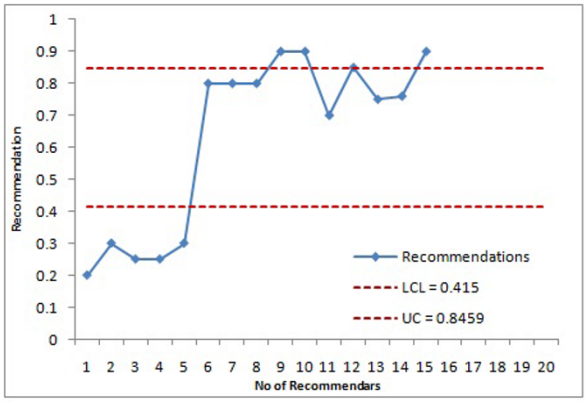

Figure 5 Detecting malicious recommendations.

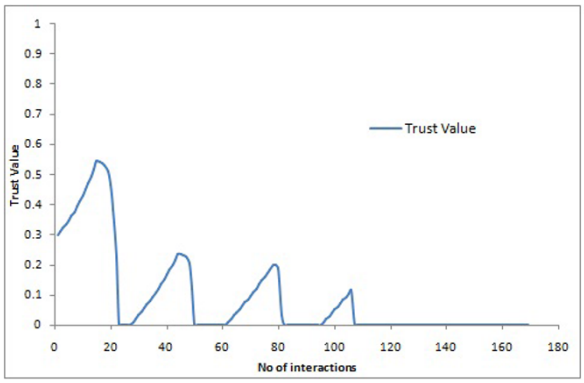

Figure 6 Strategic behavior attack.

interaction the trust value continues to decrement and after judging the behavior, model declares the entity completely distrusted and blacklist it.

6.3 Conflicting behavior attack

In this attack, malicious entities can behave inconsistently in the user domain. They can manipulate recommended trust value by performing differently to different services. For example, the attacker can always behave well with one service and behave badly to another service. These two services will develop conflicting opinions about the malicious entity. When some other service requests recommendation about the malicious entity from these services, the recommendations will not agree with each other. It will assign low recommendation trust to the recommending service believing that that it has sent dishonest recommendation. Also most of the recommendation models consider the aggregate of recommendations; in that case malicious entity can go undetected. The proposed model uses Confidence level on each recommendation is avoid this attack. Confidence level is dependent on the sensitivity of the service. Even if the malicious entity shows conflicting behavior with different services, model gives more relevance to the recommendation from services that are more sensitive than the ones that have ordinary sensitivity level. Figure 7 depicts that entity has shown conflicting behavior with different services, but its recommendation is weighted on the basis of sensitivity of the service with which it has interacted. In this scenario even though a malicious entity has shown good behavior with service G but since its sensitivity level is low ($G = 3$), recommendation from this service is given least weight.

7 Verification of system correctness

In this section, we verify that the most relevant objectives of proposed model described above to demonstrate novel theoretical concepts.

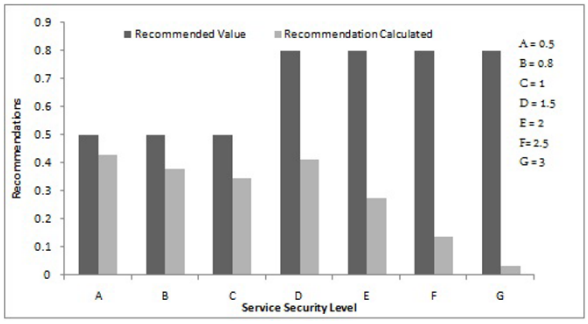

Figure 7 Detecting conflicitng behavior.

Proposition 1: The model observes the behavior of the entity and gives gradual increment/decrement initially and exponential increment/decrement subsequently before completely trusting/distrusting the entity.

Proof: The increment/decrement factor for a given positive/negative interaction is calculated by using Equations 3 and 4 respectively. Figure 8 shows trust establishment of an entity with positive and negative behaviors. The model awards gradual increment/decrement initially, however, after establishing continuity in the behaviors, the awards becomes exponential.

Proposition 2: The model judges the oscillating behavior of the entity by lowering the $T_{\max}$ of the entity.

Proof: The algorithm keeps history of entity s oscillating behavior as $\text{OnOff}_{\text{count}}$. The algorithm alters $T_{\max}$ using policy 5 each time an entity changes its behavior. Figure 9 below shows how the proposed model reacts to oscillating behavior of entity by lowering the $T_{\max}$.

Proposition 3: The model keeps history of distrustful behavior of entity and frequent distrustful behavior renders entity black listed.

Proof: Distrustful behavior of the entity is logged as *distrutCount*. The entity is rendered blacklisted if the count exceeds *distrustThreshold*. The procedure used by the model is shown in Algorithm 4:

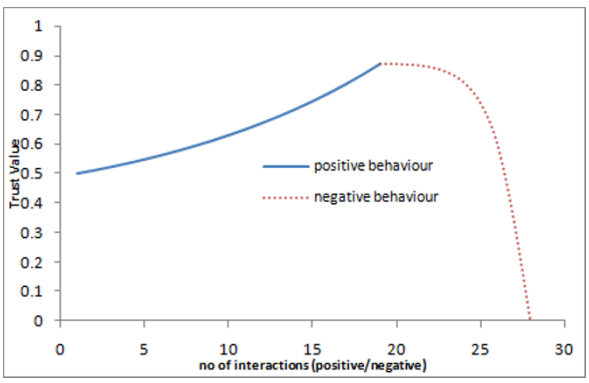

Figure 8 Growth/decline in trust value.

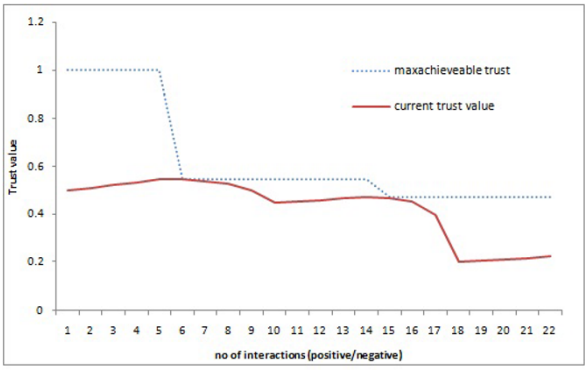

Figure 9 Effect of oscillating behaviour on maxachievable trust value.

Algorithm 4 *DisTrustEntityE*
1: *isDistrusted* = *true*
2: *distrustCount* + +
3: **if** *distrustCount* > = *distrustThreshold* **then**
4: *isBlacklisted* = *ture*
5: //No future interaction with entity
6: **end if**

Proposition 4: Reward/penalty rate after each interaction is controlled by service security level.

Proof: The slope of trust increment/decrement is dependent upon the security level of the requested service. High security level demands pro longed positive interactions to achieve maximum trust and vice versa. The most secure service will have security level 0.5. Whereas, the least secure service may have the security level equal to 3. Figure 10 depicts the effect of security level on increment/decrement of trust value.

Proposition 5: In the proposed model trust value decays with time, assigning more weights to recent observation and less weight to previous observations.

Proof: In the model, each service keeps a time stamp with its latest interaction with every entity. Each time an entity makes a request time based aging is computed on the basis of time stamp of current time and the last

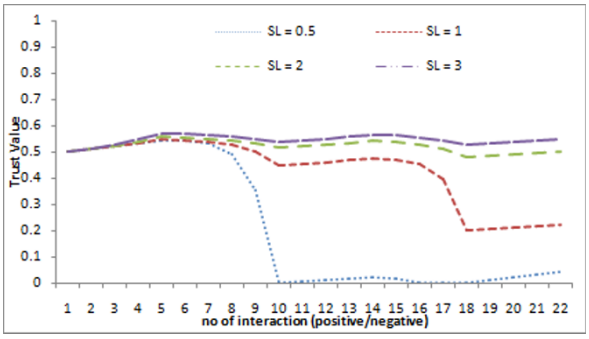

Figure 10 Effect of security level on trust value.

time the trust value was updated using Equation 3. Figure 11 shows the decay of trust value with time during the interval in which the entity did not had any new interaction with the service.

8 Comparison with other models

An important property of trust model in pervasive environment is to be adaptive, having the capability to adjust in accordance with behavioral pattern changes. The proposed model is compared with PTM [22], FTM [23] and Wang and Varadharajan [31] trust models by considering positive and negative interactions randomly (Figure 12). In proposed model, when an entity shows constant positive behavior, the increase factor grows exponentially; i.e., is gradual in beginning and then rises rapidly. A high trust value is achieved through long-term interactions with good behaviors. The increase in value depends on the number of constant positive actions. In Wang and FTM model, entity performing positively is rewarded rapidly at the beginning but gradually the trust earning rate decreases. In PTM, positive behavior has an exponential form and then a logarithmic. In all the models, as the entity continues to show constant positive behavior, they converge to maximum trust value i.e., 1. However in our proposed model, maximum achievable trust value is adjustable and controlled by the adaptable policies.

When an entity shows negative behavior, our proposed model attempts to judge whether an entity has performed negative action intentionally or unintentionally, by analyzing its sequence of interactions. Penalty factor increases with the consecutive negative behavior. Also, if an entity has history of showing deceptive behavior in past the rate of trust decline increases. Wang, FTM and PTM models punish an entity in a similar way by quickly decreasing the trust value.

Finally, if an entity shows oscillating behavior by randomly mixing both positive and negative actions, model attempts to judge the behavior and decreases the trust

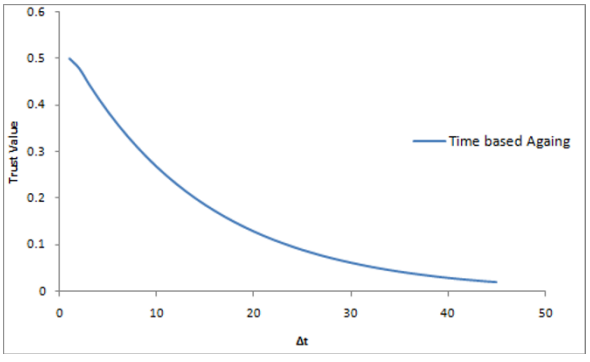

Figure 11 Time based aging.

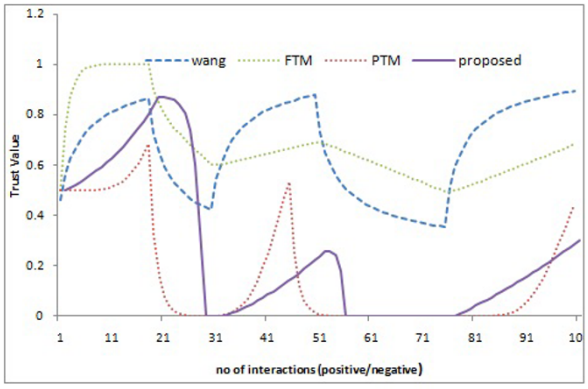

Figure 12 Comparison with other models.

value in proportion to the number of negative actions and its on/off behavior count. The final trust value is always less than the initial trust value, as the maximum achievable trust is decremented each time an entity shows an alternating behavior. In Wang model trust value is more inclined towards the last interaction value and does not consider the historical behavior. Whereas PTM takes into account the aggregate of events in order to compute the behavior pattern of an entity. Sequence of outcome of interaction has no effect on the evolution of trust thus ignoring the significance of interaction taking place at different times.

9 Conclusion

In this article, an adaptive trust and recommendation based access control architecture for pervasive environment is proposed. The proposed model handles by itself situations both in which the requesting entity has a past experience with the service and a stranger entity requesting to access the service without any identity and past interaction with the service. The main contribution of the article includes: (1) we define an adaptive trust evolution algorithm that dynamically adjust trust value according to the entity's behavior thus minimizing human involvement for the security management; (2) we introduce the concept of maximum achievable trust to regulate the susceptible behavior; (3) An adaptive policy analyzer is incorporated for strategic attack detection together with appropriate actions and controls to counter these attack; (4) motivated by human nature, we use the confidence level to judge the recommendations; and (5) also, in order filter the malicious recommendation, we introduce X-bar control charts. In addition, we have showed the effectiveness of our model, by demonstrating it to be attack resistant against Bad mouthing attack, Oscillating behavior attack and conflicting behavior attack. Our future research will also be focused on implementation of the the proposed model on smart

devices (laptops, PDAs and smart phones), to analyze the performance and optimized utilization of resources.

Author details
[1]Department of Computer Software Engineering, Military College of Signals, National University of Sciences and Technology (NUST), Islamabad, Pakistan
[2]Department of Electrical Engineering, Military College of Signals, National University of Sciences and Technology (NUST), Islamabad, Pakistan

Competing interests
The authors declare that they have no competing interests.

Received: 15 July 2011 Accepted: 27 March 2012
Published: 27 March 2012

References

1. M Weiser, *The Computer for the Twenty-First Century*, (Scientific American, 1991)
2. V Cahill, E Gray, J Seigneur, C Jensen, Y Chen, B Shand, N Dimmock, A Twigg, J Bacon, W Wagealla, S Terzis, P Nixon, G Serugendo, C Bryce, M Carbone, K Krukow, M Nielsen, Using trust for secure collaboration in uncertain environments, in *IEEE Pervasive Computing Mobile and Ubiquitous Computing*. **2**, 52–61 (2003)
3. P Robinson, H Vogt, W Wagealla, in *Some Research Challenges in Pervasive Computing, Privacy, Security and Trust within the Context of Pervasive Computing*, vol. 780. Springer, US, pp. 1–16 (2004)
4. C English, P Nixon, S Terzis, A McGettrick, H Lowe, Security models for trusting network appliances, in *5th IEEE International Workshop on Networked Appliances*, Liverpool, pp. 39–44 (Oct 2002)
5. C English, W Wagealla, P Nixon, S Terzis, A McGettrick, H Lowe, Trusting collaboration in global computing, in *1st International Conference on trust management*, Greece, pp. 136–149 (May 2003)
6. B Shand, N Dimmock, J Bacon, Trust for ubiquitous, transparent collaboration, in *1st IEEE International Conference on Pervasive Computing and Communications*, Texas, USA, pp. 153–160 (Mar 2003)
7. R He, J Niu, M Yuan, J Hu, A novel cloud-based trust model for pervasive computing, in *4th International Conference on Computer and Information Technology*, China, pp. 693–700 (Sept 2004)
8. G Ya-Jun, H Fan, Z Ping-Guo, L Rong, An access control model for ubiquitous computing application, in *2nd International Conference on Mobile Technology, Applications and Systems*, China, pp. 128–133 (Nov 2005)
9. H Jameel, LX Hung, U Kalim, A Sajjad, S Lee, Yk Lee, A trust model for ubiquitous systems based on vectors of trust values, in *7th IEEE International Symposium on Multimedia*, Irvine, USA, pp. 674–679 (Dec 2005)
10. R Bhatti, E Bertino, A Ghafoor, A trust-based context-aware access control model for web-services. Distrib Parallel Databases. **18**(1), 83–105 (2005). doi:10.1007/s10619-005-1075-7
11. MK Deno, T Sun, Probabilistic trust management in pervasive computing, in *IEEE/IFIP International Conference on Embedded and Ubiquitous Computing*, China, pp. 610–615 (Dec 2008)
12. C Hang, Y Wang, MP Singh, An adaptive probabilistic trust model and its evaluation, in *7th international joint conference on Autonomous agents and multiagent systems*, Portugal, pp. 1485–1488 (May 2008)
13. D Quercia, S Hailes, L Capra, TRULLO-local trust bootstrapping for ubiquitous devices, in *4th International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services*, Pennsylvania, USA, pp. 1–9 (2007)
14. M Blaze, J Feigenbaum, J Lacy, Decentralized trust management, in *17th IEEE Symposium on Security and Privacy*, Oakland, pp. 164–173 (1996)
15. L Kagal, T Finin, A Joshi, trust-based security in pervasive computing environments. IEEE Comput. **34**(12), 154–157 (2001). doi:10.1109/2.970591
16. L Kagal, J Undercoffer, F Perich, A Joshi, T Finin, Vigil: enforcing security in ubiquitous environments, in *Grace Hopper Celebration of Women in Computing 2002* (2002)
17. SI Ahamed, M Sharmin, A trust-based secure service discovery model for pervasive computing. Int J Commun. **31**(18), 4281–4293 (2008)
18. M Komarova, M Riguidel, Adjustable trust model for access control, in *5th international conference on Autonomic and Trusted Computing*, Norway, pp. 429–443 (June 2008)
19. F Almenarez, A Marin, C Campo, C Garcia, PTM: a pervasive trust management model for dynamic open environments, in *1st Workshop on Pervasive Security, Privacy and Trust in Conjuntion with Mobiquitous* (2004)
20. F Almenarez, A Marin, C Campo, C Garcia, TrustAC: trust based access control for pervasive devices, in *2nd International Conference Security in Pervasive Computing*, Germany, pp. 225–238 (Apr 2005)
21. F Almenarez, A Marin, D Diaz, J Sanchez, Developing a model for trust management in pervasive devices, in *3rd IEEE International Workshop on Pervasive Computing and Communication Security*, Pisa, Italy, pp. 267–272 (Mar 2006)
22. F Almenarez, A Marin, D Diaz, A Cortes, C Campo, C Garcia, Trust management for multimedia P2P applications in autonomic networking. Adhoc Netw. **9**(4), 687–690 (2011)
23. M Haque, SI Ahamed, An omnipresent formal trust model (FTM) for pervasive computing environment, in *31st Annual International Computer Software and Applications Conference*, Beijing, China, pp. 49–56 (July 2007)
24. SI Ahamed, M Haque, M Endadul, F Rahman, N Talukder, Design, analysis, and deployment of omnipresent formal trust model (FTM) with trust bootstrapping for pervasive environments. J Syst Softw. **83**, 253–270 (2010). doi:10.1016/j.jss.2009.09.040
25. E Gray, P OConnell, C Jensen, S Weber, J Seigneur, C Yong, Towards a Framework for Assessing Trust-Based Admission Control in Collaborative Ad Hoc Applications, (Technical Report, Dept. of Computer Science, Trinity College Dublin, 2002)**66**
26. A Josang, An algebra for assessing trust in certification chains, in *Proceedings of the Network and Distributed Systems Security*, San Deigo, USA, (1999)
27. A Abdul-Rahman, S Hailes, Supporting trust in virtual communities, in *33th Hawaii International Conference on System Sciences*, Mani, pp. 1–9 (Jan 2000)
28. N Dimmock, A Belokosztolszki, D Eyers, J Bacon, K Moody, Using trust and risk in role-based access control policies, in *9th ACM Symposium on Access control models and technologies*, New York, USA, pp. 156–162 (Jun 2004)
29. D Gambetta, *Can We Trust Trust?. In, Trust: Making and Breaking Cooperative Relations*, (Basil Blackwell, Oxford, 1990)
30. M Sloman, E Lupu, Security and management policy specification. IEEE Netw. **16**(2), 10–19 (2002). doi:10.1109/65.993218
31. Y Wang, V Varadharajan, Interaction trust evaluation in decentralized environments, in *5th International Conference on Electronic Commerce and Web Technologies*, Spain, pp. 144–152 (2004)

doi:10.1186/1687-1499-2012-119
Cite this article as: Iltaf *et al.*: Modeling interaction using trust and recommendation in ubiquitous computing environment. *EURASIP Journal on Wireless Communications and Networking* 2012 **2012**:119.