A Trust Management Scheme for Enhancing Security in Pervasive Wireless Networks

Mieso K. Denko and Tao Sun
Department of Computing and Information Science
University of Guelph
Guelph, Canada
denko@cis.uoguelph.ca

Isaac Woungang
Department of Computer Science
Ryerson University
Toronto, Canada
iwoungan@scs.ryerson.ca

Joel J. P. C. Rodrigues
Instituto de Telecomunicações,
University of Beira Interior
Covilhã, Portugal
joeljr@ieee.org

Han-Chieh Chao
College of Electrical Engineering and Computer Science
National I-Lan University
I-Lan, Taiwan
hcc@niu.edu.tw

Abstract—**In a pervasive wireless network, malicious nodes can initiate attacks to well-behaved nodes. This paper argues that our recently proposed probabilistic trust management scheme for pervasive computing can be used in pervasive wireless networks to provide protection against few typical attacks usually targeted at such systems. Simulation experiments are provided to assess the achievement of the stated goal. The performance metric used is the average packet loss ratio, representing the ratio of packets lost to the total packets generated in a certain time period**

Keywords- Pervasive computing; Wireless networks, Security; Probabilistic trust.

I. INTRODUCTION

With the widespread availability of inexpensive wireless solutions such as Wireless Mesh, Bluetooth, WiFi, RFID, to name a few, and the development of pervasive computing [1], the evaluation of wireless networks for pervasive computing is emerging rapidly as an exciting new research area. A pervasive wireless network can be viewed as a network intended to provide ubiquitous communication supporting voice-over-IP-based wireless coverage or data services over buildings or campuses entailing IEEE 802.11 access points. In pervasive computing environments, devices are interconnected through networks as they are in traditional networking environment, and they communicate and exchange data between each other through these networks. However, the interactions between devices are not directly controlled or monitored by human interventions. Rather, the reasons for communications are based on the predictions of human's requirements [1]. Nonetheless, a pervasive wireless networking environment can also be considered as a particular instance of a pervasive computing one. In a pervasive wireless networking environment, nodes are expected to behave autonomously when collecting, storing, processing, managing, and exchanging the data. One of the key requirements for achieving this goal is that an effective trust relationship be established among nodes in such an open mobile space, prior to data exchange (interactions) between nodes.

In a recent paper [2], we proposed a deterministic trust management scheme for pervasive computing environments, in which a single trust value is set as a parameter, allowing a device to judge another device's behavior in the presence of an interaction between them. Assuming a single trust value to assess a device's behavior may not realistically reflect the possible uneven behavior of that device, i.e., the device may turn out to be a well-behaved one or a bad-behaved one, depending on the type of services that it offers or the interactions it is involved in. In a pervasive computing environment, a device interacting with another one should be assigned multiple trust values, each representing a specific aspect of its behaviors. Under this assumption, we have recently proposed a probabilistic trust management scheme for pervasive computing environment [3]. In this trust scheme, a device uses its firsthand experience of interactions with other devices, in order to compute the trust value, either directly or indirectly. In the latter case, recommendations by other devices, as well as their trustworthiness (as judged by the device receiving those recommendations) are handled through a suitable filtering method, which in turn, is used to remedy against false recommendations. In this paper, we argue that using these intrinsic features, our trust model can effectively provide protection against attacks from badly behaved nodes in a pervasive wireless networking environment.

The rest of the paper is organized as follows. Section 2 presents some related work. Section 3 briefly introduces the aforementioned probabilistic trust model. In Section 4, we discuss few typical attacks targeted at trust management systems. In Section 5, we present simulation results, showing the effectiveness of our trust model in protecting against those attacks. Finally, Section 6 concludes our work.

978-1-4244-4148-8/09/$25.00 ©2009
This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE "GLOBECOM" 2009 proceedings.

II. RELATED WORK

Pervasive computing has emerged as a revolutionary computing paradigm [1], whose main purpose is to create and guarantee an ambient intelligence where devices embedded in the same environment provide persistent connectivity and services at all times. However, with this advantage are attached many serious challenges. For instance, in pervasive computing, designing efficient and trusted security and privacy solutions is still an open issue [1]. Of course, one of the primary steps towards establishing a security policy in such systems is to be able to ensure a descent trust relationship between devices (or other entities), both from the environment to the device, and from device to device viewpoints. To this effect, several studies have investigated security issues in pervasive/ubiquitous environments ([4], [5], [6], [7]). In these works, trust has not been used as a principal mean for designing security mechanisms. Rather, the focus has been on designing and validating security policies by other means. In contrary, few other studies have emerged on using trust as a solution for enhancing security in pervasive computing, resulting to trust model designs. In [8], the authors discussed on trust issues between devices in pervasive environment. A similar study was done in [9], resulting to a trust model where the negotiation of trust between devices is based on their security properties. Similarly, a trust model was proposed in [10] for enhancing the collaborations in mobile ad hoc networks. In paper [11], the focus was on using trust as a means for enhancing the quality of interactions and services between devices, whereas in [12], the proposed trust model relies on a communication control mechanism used to monitor the data traffic between trusted devices in the network. Other trust models were suggested, where the trust evaluation is based on establishing dynamic access rights between devices [13], devices' experience, knowledge, and recommendations [14], decomposition of the computing environment into several ad hoc networks [7], statistics and probability theories [10], [15], using recommendations as a way for enhancing trust evaluation [16], [17]. The latter series of papers also discuss typical attacks targeted against trust management systems, and provide protection methods to defend against them. In this paper, we continue the investigation of the trust management scheme introduced in [3]. The goal is to demonstrate that, in addition to providing a reliable trust relationship among nodes, this trust model can be used to protect against attacks from badly behaved nodes in pervasive wireless networking environments.

III. TARGETED TRUST MANAGEMENT SCHEME

The trust management scheme that we deal with has been introduced in [3]. Its architecture can be depicted as in Fig. 1.

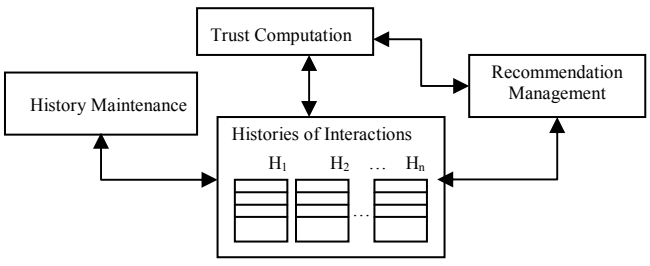

Figure 1: Probabilistic Trust Management Architecture

We refer the reader to [3] for a detailed description of this scheme. Here, we focus on its functionality, commenting mostly on the aforementioned trust model's features. In this architecture, the History of Interactions *(HI)* Module stores records on interactions between devices in a suitable data structure. The history of interactions embedded in a device A about a device B, denoted as $H_A(B)$, is a list $H_A(B) = \{H_1, \ldots, H_n\}$, kept at device A, where each entry H_i represents the trust record of a single interaction with device B, H_i is defined by thee triple $H_i = \langle e_i, s_i, d_i \rangle$, where e_i is the evaluation of the interaction, s_i is the type of interaction or service provided and d_i is the time interaction had happened. During direct or indirect computation, the *HI* module is maintained and updated by the *History Maintenance* Module. The functioning of the trust management scheme follows.

A. Trust Management Scheme Operations

In our targeted trust model, trust records are built through the *Trust Computation (TC)* process, which assigns direct trust values from observations, indirect trust values from recommendations, and dynamic properties of trust, by interacting with other modules. Indeed, trust computation happens before each interaction occurs between devices. The *TC* module selects the desired entry in the *HI* module, then decides whether to pursue with direct or indirect computation to evaluate trust values. This decision is made by computing a certain *level of confidence* that one device has in the trust evaluation on another device. This in turn depends on the amount of contextual information gathered on the interactions. If the confidence level is low, a conclusion is drawn that the confidence in the current trust information is inadequate for running a direct trust computation, thus, an indirect trust computation should be activated. Prior to running this, more information is required such as the recommendations obtained from other devices (recommenders) and the trustworthiness of these recommenders by the *TC* module prior to accepting and collecting their recommendations. This additional information is gathered through interactions between the *TC* module and the *Recommendation Management* module. Once the above judgments on recommenders and their recommendations are completed, the indirect trust computation is activated by the TC module and run with the help of accepted "trusted" recommendations, provided by the *Recommendation Management* module, in addition to records of known experience on history of interactions it received from the *HI* module. During this process, an iterative filtering method [3] is employed at a second judgment level by the *TC* module, to filter out among honest recommenders while ensuring that their recommendations are still as much accurate as possible. In the design of this iterative filtering method, a threshold value between [0,1] is used as a decision factor to accept or discard a recommendation from a given device.

In our trust model, the dynamic property of trust is reflected in the way the *TC* module computes trust values that represent as accurately as possible the device's behaviors. To illustrate this point, let us assume that the pervasive computing environment is composed of three devices *P*, *Q*, and *R*. Also, assume that during its interactions with device *P*, device *Q*

978-1-4244-4148-8/09/$25.00 ©2009
This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE "GLOBECOM" 2009 proceedings.

behaved well 10 times, then badly 10 times, i.e., in $H_P(Q)$, there should be 10 entries with value 1, followed by 10 entries with value 0. Similarly, during its interactions with device P, device R behaved badly 10 times, then well 10 times, i.e., in $H_P(R)$, there must be 10 entries with value 1, followed by 10 entries with value 0 (see Table 1). From Table 1, one can observe that currently Q behaves badly while R behaves well. Since the trust value is predicted based on the current behavior of a device, it follows that the trust value that P perceives about Q must be lower than the one that P perceives about R, i.e. $T_p(Q) < T_p(R)$. Now, assume that no method is applied on the result of interactions and each result of an interaction is treated equally. When P runs the trust computation on Q and R, it will be using the parameters $n_s^Q = n_u^Q = n_s^R = n_u^R = 10$, where n_s^R and n_u^R represent the number of satisfying and unsatisfying interactions respectively, in the device R's recommendation. This will result in the same trust values for Q and R, i.e. $T_p(Q) = T_p(R) = \frac{1}{2}$, which is a contradiction. Thus, these trust values do not accurately reflect the current behaviors of devices.

TABLE 1: EXAMPLE OF HISTORIES OF INTERACTIONS

$H_P(Q)$			$H_P(R)$		
d	***e***	***s***	***d***	***e***	***s***
1	1	1	1	0	1
……	……	……	……	……	……
10	1	1	10	0	1
……	……	……	……	……	……
20	0	1	20	1	1

This example shows that treating the results of interactions equally may result in a misleading outcome. Instead, our trust model considers that the results of recent interactions are more important than those of older interactions because they represent the current behaviors of the device. To accurately reflect the device's behaviors, a weighting method [3] is implemented to assign weights to records of interactions based on when these interactions have occurred.

Finally, our trust model can be used in pervasive wireless networks, for instance, to protect the routing process when malicious nodes have entered the network. To this effect, a trust value of a node can be associated with the action of forwarding the packets or making recommendations to other nodes. When a source S wishes to establish a route to a destination D, it will find multiple routes to D, and determine the packet-forwarding trustworthiness of all nodes (including malicious ones) on these routes based on recommendations from other nodes or on its own trust record history. Then, S will select the 'best' trustworthy path for sending the packets. Afterwards, trust records of all nodes will be updated.

B. Illustrating the Weighting Method

To illustrate the effectiveness of our weighting method, let's consider our previous example and recalculate the trust values for devices Q and R. Let's assume that the current time is $t_{cur} = 21$ and the factor $w = 0.7$. Using formulas presented in [3], the weights for entries in $H_P(Q)$ and $H_P(R)$ are obtained as:

$$n_s^Q = 0.064 \text{ and } n_u^Q = 2.27 \text{ (for device } Q\text{)} \quad (1)$$

$$n_s^R = 2.27 \text{ and } n_u^R = 0.064 \text{ (for device } R\text{)} \quad (2)$$

It follows that $T_P(Q)$ the trust value that P puts on Q (respectively $T_P(R)$ the trust value that P puts on R) is:

$$T_p(Q) = \frac{n_s^Q + 1}{n_s^Q + n_u^Q + 2} = \frac{0.064 + 1}{0.064 + 2.27 + 2} = 0.246 \quad (3)$$

$$T_p(R) = \frac{n_s^Q + 1}{n_s^Q + n_u^Q + 2} = \frac{2.27 + 1}{2.27 + 0.064 + 2} = 0.754 \quad (4)$$

Clearly, our weighting method yields $T_P(Q) < T_P(R)$. This relation reflects the fact that the trust value that P puts on Q should be less than the trust value that it puts on R since device Q behaved badly and device R behaved well, which is understandable. Detailed results are provided in the simulation experiments Section to further illustrate the benefits of using this weighting technique.

IV. FEW TYPICAL ATTACKS TARGETED AT TRUST MANAGEMENT SYSTEMS

In trust management schemes that use recommendations, there exist some typical attacks that may be launched by selfish and malicious devices [16]. Few of these attacks have been identified in [3], referred to as (1) *Low quality interaction attacks* – this type of attacks includes denial of service, delaying of service, dropping packets, to name a few, (2) *False Recommendation Attacks* – this attack occurs when malicious recommenders are involved in the indirect trust computation process, providing false recommendations to frame up well-behaved devices or even to boost the trust values of other malicious peers, (3) *Dynamic Behaviors Attacks* – here, malicious devices behave well and badly in alternate time periods, hoping that they can remain undetected while causing damage. In [3], a qualitative analysis is provided to discuss how the trust model introduced here can deal with the above attacks. In the sequel, simulation experiments are presented to validate this analysis.

V. PERFORMANCE EVALUATION

The effectiveness of the proposed trust management scheme in improving the quality of interactions and in providing trust between devices in a pervasive computing environment has been proved in [3]. In this section, we demonstrate through simulation experiments that our trust management scheme can also be used to effectively address the mentioned typical attacks. In our experiments, we use the following varying parameters: (1) length of simulation time, (2) number of interactions between devices, proportion of false recommenders – which defines the number of service

978-1-4244-4148-8/09/$25.00 ©2009
This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE "GLOBECOM" 2009 proceedings.

requesters (SRs) that provide false recommendations over the total number of SRs, and (3) proportion of malicious devices – which defines the number of malicious service providers (SPs) over the total number of SPs. We change the values of these parameters, under various scenarios, to investigate the influence they have on the average packet loss ratio. The following assumptions are made: (a) the interaction time follows an exponential distribution. During the interaction, a Poisson process is used to generate the packet arrival time. If no service provider (SP) is idle, the service requester (SR) will wait for a period of time and will attempt an interaction again, (b) the waiting time and the interval time between two interactions follow an exponential distribution. The values of these simulation parameters are: average interaction time (20 seconds), average packet arrival time and average waiting time (1 second each), and average interval time (3 seconds).

The simulation environment is composed of 20 service providers (SPs) and 80 service requesters (SRs). It is assumed that (1) SPs do not interact with each other and do not provide recommendations, (2) Each SP can serve only one SR at a time, (3) SRs can interact between them for recommendations purpose, (4) SRs can interact with all SPs for packets relay purpose, and in this capacity, they have the same expectations on their interactions with SPs, (5) Low quality interaction attack only resumes on the action of dropping packets.

A. Protection Against Low Quality Interaction Attacks

In this experiment, we study whether a device could use our trust model to find appropriate devices to interact with and detect malicious devices efficiently, and whether our trust model can protect the device against the chosen low quality interaction attack (dropping packets in this case). We assume that the behaviors of devices are static, out of the investigated SPs, 20% of them are malicious ones that drop packets during an interaction. We set three scenarios: (1) *the scenario without using trust management* – where devices do not use a trust management scheme. When a SR needs to interact with a SP, it randomly chooses an idle SP regardless of the previous results of interactions with this device, (2) *the scenario using trust management without recommendations* – where the SRs use the trust model when choosing a SP. However, SRs use the trust value obtained only through direct trust computation to estimate whether a SP is trustworthy for interaction (since no recommendation is allowed), (3) *The scenario using trust management with recommendations* - where we apply the trust management scheme with both direct and indirect trust computations to each SR. When a SR attempts to interact with a SP, it first investigates whether direct computation is possible. If the confidence for direct trust computation is inadequate, it request recommendations from other SPs to run an indirect trust computation. Here we assume that all devices make honest recommendations.

The results of our experiments are depicted in Fig. 2. It is observed that the average packet loss ratio without trust management (*no-trust*) remains at a high level, and the average packet loss ratios with trust management (*without-recomm* and *trust-recomm*) decrease with increasing simulation time. In the first scenario, the SRs do not judge the SPs. For each interaction, they randomly choose an idle service provider to interact with. Hence, no matter how long the simulation time is, the *no-trust* average packet loss ratio stays at a high value. In the second scenario, it is observed that with increasing simulation time, the average packet loss ratio of the *without-recomm* scenario becomes lower. When the simulation time is short, the average packet loss ratio stays high. The reason is that at this phase, SRs do not have enough records of interactions to directly compute trust values with adequate confidence. Because they cannot make the right judgment on SPs, they are not always able to avoid interactions with malicious SPs. However, with an increase in simulation time, SRs can accumulate enough records of interactions, so they are able to obtain trust values through direct trust computations. Therefore, malicious SPs can be correctly detected and interactions with them can be avoided. In the third scenario, it is observed that at the initial phase of simulation, the average packet loss ratio without recommendations (*without-recomm*) and that with recommendations (*trust-recomm*), are at similar levels. This is justified by the fact that at the initial phase of the simulation, all SRs have only few records of interactions. Although recommendations can be provided by other SRs, the trust information available at that time period isn't enough for trust computation. With an increase in simulation time, the SRs accumulate more records of interactions. In this case, the recommendations from other SRs will help a SR to run the indirect trust computation. Therefore, when the simulation time is longer, the average packet loss ratio for the *trust-recomm* scenario is much lower than that of the *without-recomm* scenario. However, when the simulation time continues to increase, the difference between the two values becomes smaller. The reason for this is that at this phase, most SRs have accumulated enough records of interactions to run direct trust computations, and SRs can use trust values to choose SPs with good performance to interact with.

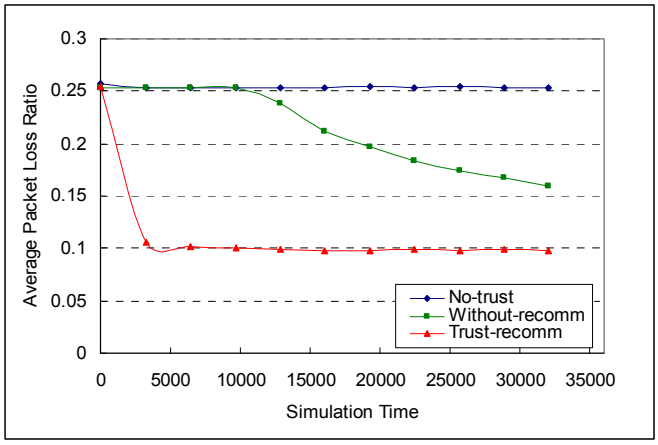

Figure 2: Simulation Time under Varying Average Packet Loss Ratio.

B. Protection Against Dynamic Behaviors Attacks

In this experiment, we investigate whether our trust management scheme can make accurate trust computation

978-1-4244-4148-8/09/$25.00 ©2009
This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE "GLOBECOM" 2009 proceedings.

when the behaviors of some devices are dynamic. By dynamic behavioral attacks, we mean the environment that contains some malicious devices whose behaviors change overtime. These devices may behave well in the first several interactions and behave maliciously then after. By doing this, they accumulate good records in other devices' history of interactions. Then they initiate attacks by behaving badly during the next interactions, hoping that those good records of interactions that they accumulated previously can cause other devices to run inaccurate trust computations on their current behaviors, so that their attacks will be difficult to detect. By conducting this experiment, we study whether our proposed weighting method can be used to protect the devices against the dynamic behaviors attacks. Here, a device D_0 interacts with another device D_1 that launches a dynamic behaviors attack. In the first 10 interactions, D_1 behaves well. In the following interactions, D_1 behaves badly by dropping packets. We set two scenarios: (1) *the scenario without weighting method* – where a SP D_1 behaves alternatively. Device D_0 is not subject to any protection against this type of attack. We study the influence of this type of attack on the accuracy of trust computations, (2) *the scenario with weighting method* – where we apply the weighting method, with a weighting factor of 0.99. The results are depicted in Fig 3 and Fig. 4.

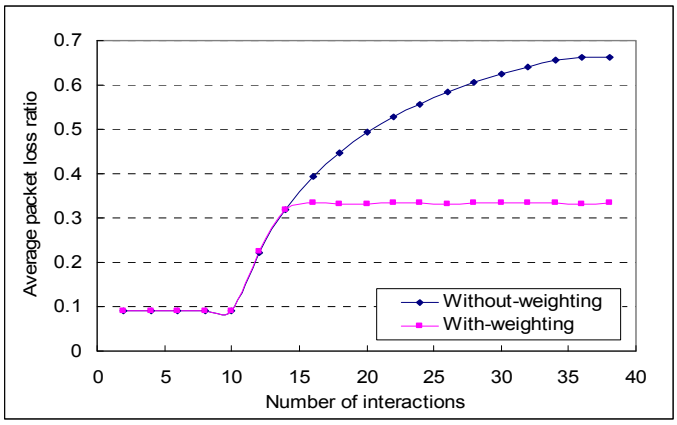

Figure 3: Average Packet Loss Ratio under Varying Number of Interactions.

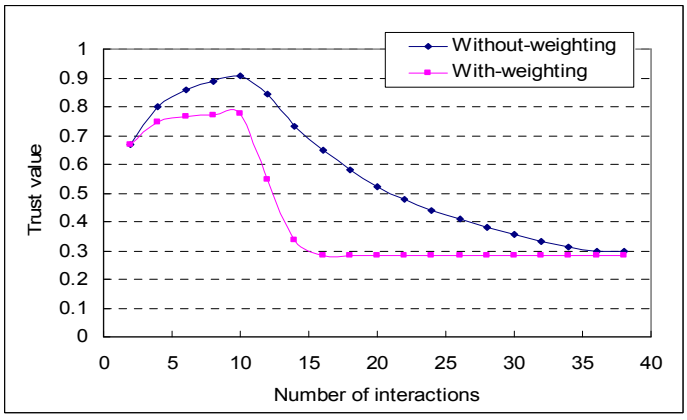

Figure 4: Trust Value under Varying Number of Interactions.

It is observed that in the first 10 interactions, the average packet loss ratios of both scenarios are low (Fig. 3), and the trust values of both scenarios are increasing (Fig. 4) since D_1 behaved well. After 10 "good" interactions, D_1 launches the attack by dropping packets. In the first scenario (*without-weighting*), it is observed that the trust value decreases after the first 10 interactions, but the decreasing rate is very low, and for D_0, it took over 30 more interactions to reduce the trust value about D_1 below the specified threshold (see Fig. 4), i.e., to detect that D_1 is a malicious device. During this time period, the interactions did continue between D_0 and D_1, and the packet loss ratio did continue to increase (Fig. 3). This observation shows that in the first scenario, device D_0 is not protected against the attack and the attacker D_1 has achieved a dynamic behavior attack. In the second scenario (*with-weighting*), it is observed that the trust value decreases after the first 10 interactions, and the decreasing rate is fast (see Fig. 4). In fact, for D_0, it took only 5 interactions to get the trust value about D_1 below the specified threshold, i.e. to detect D_1 as a malicious device. After that time period, the interaction between D_0 and D_1 has no longer continued, which explains why the average packet loss ratios for the second scenario are constant in Fig.3 for that specified time period. These facts show that by applying the weighting method, our proposed trust management scheme has quickly detected the malicious device D_1, and, has effectively protected the device D_0 against the dynamic behaviors attack.

C. *Protection Against False Recommendation Attacks*

In this experiment, we investigate the capability of our trust management system to protect the device against false recommendations attacks. In fact, we want to study how our proposed iterative filtering method reacts when changing the proportion of SRs that provide false recommendations (assuming that among the available 80 SRs, some provide false recommendations whereas others provide honest recommendations). In that effect, we assume that (a) the number of devices that provide false recommendations increases from 4 to 36, i.e., in proportion of 5% and 45% of the SRs respectively, (b) 4 out of the 20 SPs are malicious devices that drop packets during the interactions, and (c) false recommendation providers provide false positive recommendations about malicious devices, i.e., when providing recommendations about malicious devices, they recommend a high value for the number of satisfying interactions and 0 as the number of unsatisfying interactions. We set two scenarios: (1) *the scenario of false recommendation attacks without protection* – where the trust model does not differentiate the recommendations when running indirect trust computations. That is all recommendations are accepted without judgment. Here, the goal is to investigate whether the performance of our trust management scheme can be influenced by false recommendations, (2) *the scenario with protection against false recommendation attacks* – where the trust model uses the iterative filtering method to evaluate the accuracy and trustworthiness of recommendations when running indirect trust computations. Here, the goal is to investigate whether the

978-1-4244-4148-8/09/$25.00 ©2009
This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE "GLOBECOM" 2009 proceedings.

iterative filtering method can effectively protect devices against false recommendations attacks.

The results of this experiment are depicted in Fig. 5. It is observed that when the number of false recommenders is no more than 16 (i.e., less than 20% in proportion), the average packet loss ratios using the filtering method (*with-filtering*) remain at a level lower than 0.1. However, when the proportion of false recommenders is more than 20%, the average packet loss ratio increases gradually, but is still lower than that obtained without using the filtering method (*no-filtering*). When the proportion of false recommenders is more than 40%, the average packet loss ratios of the two scenarios are almost the same. On the other hand, the average packet loss ratio of the *no-filtering* case remains high with the increase in proportion of false recommenders. In the first scenario, trust computations are influenced by false recommendations. The inaccurate trust computations have allowed the SRs to choose malicious SPs for interactions, leading to high values of the average packet loss ratios. In the second scenario, the proposed iterative filtering method is applied. Based on the results of the experiment, one can conclude that when the proportion of false recommenders is no more than 20%, the iterative filtering method works quite well. Even when this proportion is about 30%, the performance of the filtering method is still acceptable. This helps the SRs to avoid the influence of false recommendations and to find the trustworthy SPs to interact with. However, when the proportion of false recommenders increases to more than 35%, the iterative filtering method does not work effectively. This is understandable because during the design of the filtering method, it was assumed that the majority of devices in the environment make honest recommendations.

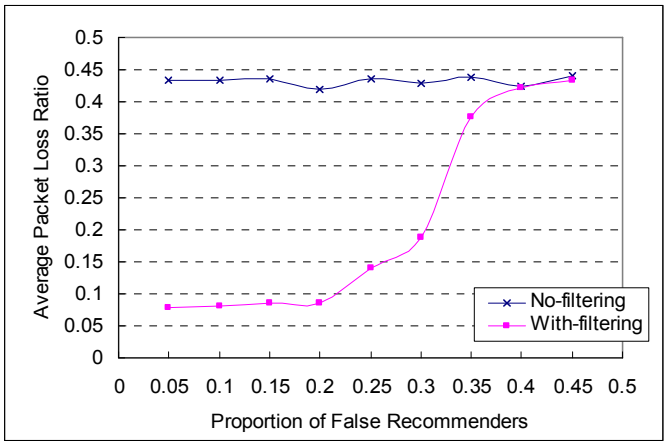

Figure 5: Average Packet Loss Ratio under Varying False Recommenders.

VI. CONCLUSION

In this paper, we have investigated a recently proposed trust model. We have established that in addition to improving the quality of interactions and providing effective trust between devices in a pervasive computing environment, including pervasive wireless networking environment, our trust model can also effectively deal with three types of typical attacks targeted at trust management systems, namely, low quality interaction attacks, false recommendation attacks, and dynamic behaviors attacks. An interesting extension of this work is to improve the method for protecting devices against false recommendations. This requires the development of adaptive filtering methods and collusion detection models.

REFERENCES

[1] D. Saha, and A. Mukherjee, "Pervasive computing: a paradigm for the 21st century," In *IEEE Computer*, IEEE Computer Society Press, pp. 25-31, 2003.

[2] M. K. Denko, S. Tao, I. **Woungang, "***Deterministic trust management in pervasive computing*", .Journal of Mobile Multimedia, vol.5, no.1, 2009.

[3] M. K. Denko, T. Sun, "Probabilistic trust management in pervasive computing," In *Proc. IEEE/IFIP Int. Conference on Embedded and Ubiquitous Computing (EUC'08)*, pp.610-615, Dec. 17-20, Shangai, China, 2008.

[4] R. Campbell, J. Al-Muhtadi, P. Naldurg, G. Sampemane, M. Mickunas. "Towards security and privacy for pervasive computing," *LNCS*, vol. 2609, pp. 77-82, 2003.

[5] A. Boukerche, Y. Ren, "A trust-based system for ubiquitous and pervasive computing environments," *In Computer Communications*, 31 (2008), pp. 4343-4351.

[6] A. Tripathi, T. Ahmed, D. Kulkarni, R. Kumar, K. Kashiramka, "Context-sased secure resource access in pervasive computing environments," *In Proc. of the 2nd IEEE Annual Conference on Pervasive Computing and Communications Workshops*, pp. 159-163, 2004.

[7] S. T. Wolfe, S. I. Ahamed, M. Zulkernine, "A trust framework for pervasive computing environments*," In Proc. of IEEE Int.l Conference on Computer Systems and Applications*, pp.312-319, 2006.

[8] M. Langheinrich, "When trust does not compute - the role of trust in ubiquitous computing," In Proc. of Privacy Workshop in Ubicomp'03, 2003.

[9] K. Ranganathan, "Trustworthy pervasive computing: the hard security problems," *In Proc. of the Second IEEE Annual Conference on Pervasive Computing and Communications Workshops*, pp. 117-121, 2004.

[10] C. T. Nguyen, O. Camp, S. Loiseau, "A Bayesian network based trust model for improving collaboration in mobile ad hoc networks," In Proc. IEEE International Conference on Research, Innovation and Vision for the Future, pp. 144-151, 2007.

[11] L. McNamara, C. Mascolo, L. Capra, "Trust and mobility aware service provision for pervasive computing," *Proc. 1st Intern. Workshop on Requirements and Solutions for Pervasive Software Infrastructures*, pp. 603-610, 2006.

[12] Z. Liu, D. Xiu, "Agent-based automated trust negotiation for pervasive computing." *Proc. of the 2nd International Conference on Embedded Software and Systems*, pp. 300-309, 2005.

[13] L. Kagal, T. Finin, and A. Joshi, "Trust-based security in pervasive computing environments," *IEEE Computer*, vol. 34, issue 12, pp. 154-157, 2001.

[14] S. Yin, I. Ray, I. Ray, "A trust model for pervasive computing environments," Proc, of CollaborateCom2006, pp. 1-6, 2006.

[15] D. Quercia, S. Hailes, L. Capra, "B-trust: Bayesian trust framework for pervasive computing," *LNCS Science*, vol. 3986, pp. 298- 312, 2006.

[16] Y. L. Sun, Z. Han, W. Yu, K. J. R. Liu, "A trust evaluation framework in distributed networks: vulnerability analysis and defense against attacks," In *Proc. of INFOCOM 2006*, pp. 1-13, 2006

978-1-4244-4148-8/09/$25.00 ©2009
This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE "GLOBECOM" 2009 proceedings.