International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:1, No:3, 2007

Understanding and Measuring Trust Evolution Effectiveness in Peer-to-Peer Computing Systems

Farag Azzedin and Ali Rizvi

***Abstract*—** In any trust model, the two information sources that a peer relies on to predict trustworthiness of another peer are direct experience as well as reputation. These two vital components evolve over time. Trust evolution is an important issue, where the objective is to observe a sequence of past values of a trust parameter and determine the future estimates. Unfortunately, trust evolution algorithms received little attention and the proposed algorithms in the literature do not comply with the conditions and the nature of trust. This paper contributes to this important problem in the following ways: (a) presents an algorithm that manages and models trust evolution in a P2P environment, (b) devises new mechanisms for effectively maintaining trust values based on the conditions that influence trust evolution , and (c) introduces a new methodology for incorporating trust-nurture incentives into the trust evolution algorithm. Simulation experiments are carried out to evaluate our trust evolution algorithm.

***Keywords*—** P2P, Trust, Reputation, Incentives.

I. Introduction

THE *peer-to-peer* (P2P) computing is one of the technologies that is having a significant impact on the way Internet-scale systems are built. It is well established for applications such as file sharing (e.g., Gnutella [16] and KaZZa[20]) and parallel distributed computation (e.g., SETI@home [1]). The popularity of P2P computing has prompted the research community to examine several aspects of it. One aspect is to extend P2P computing to host a wider variety of applications. Several approaches including the following are investigated to achieve this goal: (a) constructing generalized P2P overlays [5], (b) using P2P overlays as resource provisioning systems for resource management infrastructures such as Grid systems [19], and (c) hybrid systems that combine P2P and Grid computing techniques [9].

One of the issues that is common to all these approaches is trust [4], [17], [9], [8]. Existing trust models can be classified according to: (a) the components used to compute trust and (b) the mechanism used to aggregate recommendations. Trust models compute trust based on two components: direct trust and/or reputation. If reputation is used to compute trust, then recommendations can be aggregated by taking a weighted average, a majority voting, and so on.

In a trust model, when peer x decides to undertake a transaction with peer y, x may do so based upon two sources of information, namely the reputation of y and the direct trust (i.e., experience) that x has with y in their past transactions. During a transaction, x may choose to monitor the transaction. The monitoring results can be used to evolve the behavior trustworthiness of y in this particular transaction and can be used to evolve the trust level that x has in y. In the trust evolution process, the current transaction evaluation is aggregated with x's past experience with y to compute x's overall direct trust with y. The assessment of direct experience as well as the recommended trust is a continuous process that evolves x's trust in y and also x's trust in its recommenders [4], [18].

In this paper, we are concerned with the process of evolving x's direct experience with y. Trust evolution is an important mechanism in any trust system [4], [18] and it significantly contributes to y's trust prediction for future transactions.

A. Motivation

In any trust model, the two information sources that x relies on to predict y's trustworthiness are essentially a result of trust evolution. Direct experience or first hand information is a product of trust evolution. Reputation or second hand information is also a product of trust evolution since the recommenders give recommendations based on their direct experience with y. Therefore, trust evolution plays a crucial role in the heart of trust modeling. A trust evolution algorithm should create opportunities and allows for growth. We quote from [15] "if heavy regulations is capable of eradicating overtures of trust, and of driving out opportunities for trusting relationships, then it is capable of doing great harm". In addition, a trust evolution algorithm should comply with the conditions which influence trust evolution. Sociologists [13], [14] view trust as a hard to build and easy to destroy. Others such as [12] studied the conditions in which trust declines and states that trust declines with: (a) decreased interaction frequency, (b) increased interactions of "outsiders" (i.e., increased interaction with people whom we do not know).

Since 1950s, trust and trust relationships have been the subject in the offline world in many disciplines including philosophy, sociology, psychology, and management [15], [10], [7]. While trust has been identified as a critical factor in many offline non-technical human endeavors, researchers are just beginning to study it in the context of technology [17], [4], [2] and there is a lack of standards towards and about online trust [7], [10]. Trust evolution algorithms, in particular, received

Manuscript received March 27, 2007. This work was supported in part by the Deanship of Scientific Research (DSR) at King Fahd University of Petroleum and Minerals (KFUPM) under Grant JFRG-2006-16.

Dr. Farag Azzedin is an Assisstant Professor with KFUPM, P.O. Box 1290, Dhahran, Saudi Arabia. Dr. Farag Azzedin is the corresponding author. Phone: 03-860-3431; fax: 03-860-2174; e-mail: fazzedin@kfupm.edu.sa.

Ali Rizvi is a Graduate Student with the Department of Information and Computer Science at KFUPM; e-mail:alirizvi@kfupm.edu.sa.

little attention and the proposed algorithms in the literature do not comply with the conditions and the nature of trust.

B. Contributions

This paper contributes to this important problem in the following ways: (a) presents an algorithm that manages and models trust evolution in a P2P environment, (b) devises new mechanisms for effectively maintaining trust values based on the conditions that influence trust evolution , and (c) introduces a new methodology for incorporating trust-nurture incentives into the trust evolution algorithm.

II. Related Work

The authors in [3] used an *exponential weighted moving average* (EWMA) filter for trust evolution. One of the drawbacks of this scheme is that it returns high estimates despite periodic occurrences of low values in a sequence of trust values (i.e., a peer can periodically cheat and still maintain a high trust level). The EWMA filter produces an estimate using $O_t = \omega O_{t-1} + (1-\omega)O_c$, where $0 \leq \omega \leq 1$, O_t is the newly generated estimate, O_{t-1} is the prior estimate, and O_c is the newly generated observation. If ω is large, the filter resists rapid changes in individual observations and said to provide stability. For low ω values, the filter is able to detect changes quickly and said to be agile.

In [11], [6], these filters were combined to create a *flipflop* filter. A *flipflop* filter consists of two EWMA filters. One is agile with ω of 0.1 and the other is stable with ω of 0.9. A controller makes a decision to select between the two filters such that it selects the agile filter when possible, but falls back to the stable filter when new observations are unusually noisy.

In AzM04, a variation of the *flipflop*, filter can be applied to trust evolution where the agile filter is activated as soon as we detect a drop in value of the trust parameter beyond an acceptable threshold from the previously estimated value. The agile filter quickly downgrades the estimate. For the subsequent estimates, we switch back to the stable filter assuming that the trust parameter does not experience any further depreciations. One of the drawbacks of this filter is that it does not penalize those peers that continue to periodically cheat.

In [4] a further and recent modification of the flipflop is developed to obtain *weighted modified flipflop* (WMFF) to take periodic cheating into considerations. This trust evolution algorithm uses the total number of untrustworthy transactions to deduce O_c's effect on O_t. The more the untrustworthy transactions, the smaller will be the effect of O_c in improving the evolved trust. One of the drawbacks of this algorithm is the lack of incentives to nurture and encourage trustworthy behavior especially among the untrustworthy peers.

III. Proposed Trust Evolution Algorithm

A. Overview

In a trust model, the algorithm chosen to update the trust parameters is very important for the following reasons. First, depending on the nature of the parameter, a specific update algorithm might be preferred over another. For example, we can not use a flip flop filter, which applies an agile filter as soon as quick changes are detected. Although an agile filter is appropriate for a quick drop in the trust level, an agile filter is not appropriate for a quick increase in the trust level because trust is difficult to build and easy to lose [17]. Second, using a specific update algorithm might not be suitable in detecting untrustworthy peers. For example, using EWMA would return high estimates despite the periodic occurrence of low values in the sequence (i.e., a peer can periodically cheat and still maintain a high trust level).

B. Proposed Algorithm

We designed a trust evolution algorithm called *time-aware incentive-based weighted flipflop* (TIWFF). This algorithm complies with the conditions and the nature of trust explained in Section I-A. TIWFF devises new mechanisms for effectively maintaining trust values based on the conditions that influence trust evolution and introduces a new methodology for incorporating trust-nurture incentives. Incentive mechanisms are introduced to encourage untrustworthy peers to improve their behavior after having had an untrustworthy history.

Our proposed algorithm works as follows. If the new trust observation (i.e., O_c) is worse than the prior trust estimate (i.e., O_{t-1}), the agile filter is used to evolve trust because we want to detect changes quickly. This is shown in lines 7 and 8 of Figure 1. When the new trust observation (i.e., O_c) is better than the prior trust estimate (i.e., O_{t-1}), the stable filter is used to evolve trust in case there have not been any untrustworthy transactions or they are older than a threshold amount of time. This case is shown in lines 9 to 12 of Figure 1.

```
TIWFF-Trust Evolution Algorithm($O_{t-1}$, $O_c$, $num_{ut}$, $t_{ut}$, $t_t$)
(2)  $O_{t-1}$ ;; the prior trust estimate
(3)  $O_c$ ;; the new trust observation
(4)  $num_{ut}$ ;; the number of untrustworthy transactions
(5)  $t_{ut}$ ;; the time of the last untrustworthy transaction
(6)  $t_t$ ;; the time duration after which a transaction is
     considered to be old
(7)  if $O_c \leq O_{t-1}$
(8)      use agile filter
(9)  else
(10)     $\alpha_{ut} =$ abs $(t_c - t_{ut})$
(11)     if $num_{ut} == 0$ or $\alpha_{ut} == t_t$
(12)         use stable filter
(13)     else
(14)         $\beta_a = \alpha_{ut} \; / \; t_t$
(15)         $\beta_i = (1 - \beta_a) * (num +_{ut} -1) + 1$
(16)         Use stable filter with $\beta_i$
(17) end
```

Fig. 1. Time-aware incentive-based trust evolution algorithm.

In the other case, when there are untrustworthy transactions that are not too old, TIWFF makes it more difficult to improve trust, but an incentive is given in the form of reducing this difficulty level as the untrustworthy transactions become old gradually. The age (i.e., α_{ut}) of the last untrustworthy transaction is computed as shown in line 10.

The stable filter is used with an incentive factor to vary the difficulty of improving trust, as shown in line 15. The incentive factor (i.e., β_i) is made to vary from the number of untrustworthy transactions (i.e., num_{ut}) to 1. The computation of β_i involves two components: the age ratio (i.e., β_a) and num_{ut} as shown in line 15. Age ratio is the ratio of the age of the last untrustworthy transaction to t_t as shown in lines 6 and 14. As illustrated in line 15, we take $1 - \beta_a$ instead of age ratio because we want to decrease the punishing effect of untrustworthy transactions as the age increases. $1 - \beta_a$ decreases from 1 to 0 as age increases, hence it can be used to model the incentive factor to vary from num_{ut} to 1. The subtraction and addition of 1 in the equation in line 15 make sure that the incentive factor is mapped over the range num_{ut} and 1 and doesn't exceed on either limit of this desired range. The variation of the incentive factor and other variables with respect to changes in incentives and punishments are illustrated in Figure 2 and Figure 3.

Our proposed trust evolution algorithm is time-aware as well as untrustworthy-behavior-aware. The more recent the last untrustworthy transaction, the more difficult to improve trust; and the older the last untrustworthy transaction, the easier it will become to improve trust. But this ease of improving trust never exceeds the stable algorithm and gets clamped when it reaches the level of the stable algorithm.

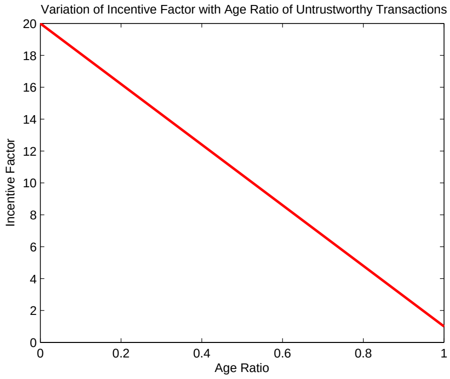

Fig. 2. Variation of incentive factor with age ratio., $num_{ut} = 20$.

IV. Performance Evaluation

In this section, we discuss the results from our simulation experiments to analyze the performance of the proposed trust evolution algorithm.

A. Overview

We conducted a series of simulation studies to examine various properties of the proposed trust evolution algorithm (i.e., TIWFF). Some of the performance measures of TIWFF that we investigated in this study are: (a) the ability to preserve the conditions of trust. That is, trust is hard to build and easy to destroy, trust declines with decreased interaction frequency, and trust declines with increased interactions with malicious peers , (b) the ability to correctly detect peers that continue to periodically cheat, and (c) the ability to nurture and encourage trustworthy behavior in the P2P environment.

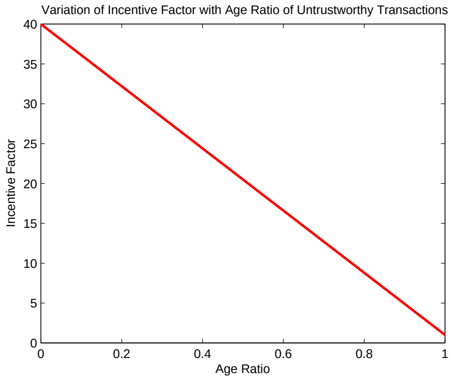

Fig. 3. Variation of incentive factor with age ratio., $num_{ut} = 40$.

TABLE I
Description of the trust levels.

Equivalent numerical value	Description
1	very untrustworthy
2	untrustworthy
3	medium trustworthy
4	trustworthy
5	very trustworthy

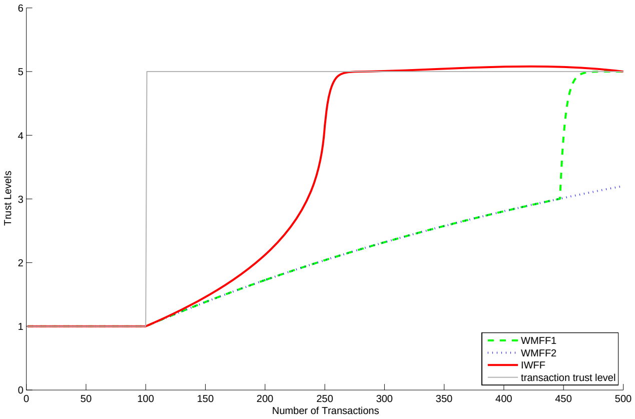

Fig. 4. Trust Evolution using WMFF1, WMFF2 and TIWFF for Scenario 1.

B. Simulation Model and Setup

Behavior trust is quantified by a dynamic parameter called *trust level* (TL) that ranges from *very untrustworthy* to *very trustworthy*, and which is represented by a numeric range [1..5] as shown in Table I.

In this simulation study, we compare 4 trust evolution algorithms using a setup where two peers are directly connected to each other such that peer x it interacting with peer y. Transactions are simulated between x and y. The newly generated observation by x regarding y is 5 if y

is trustworthy and 1 if y is untrustworthy. The number of transactions simulated is taken to be 500. In this simulation, we compare two variations of WMFF [4] trust evolution algorithms namely WMFF1 and WMFF2 with our proposed time-aware incentive-based weighted flipflop (TIWFF) algorithm. The difference between WMFF1 and WMFF2 is in their awareness and unawareness of trustworthiness zones. WMFF1 is aware of trustworthy ($3 \leq$ trust ≤ 5) and untrustworthy zones ($1 \leq$ trust < 3) and applies number-of-untrustworthy-transactions-based-punishment only when the target peer is in the untrustworthy zone. WMFF2 does not distinguish between these zones. We consider four different scenarios in terms of behavior displayed by target peer in 500 transactions: (a) **Scenario 1:** Initially untrustworthy for 100 transactions, later trustworthy, (b) **Scenario 2:** Initially trustworthy for 100 transactions, later untrustworthy, (c) **Scenario 3:** Initially untrustworthy for 100 transactions, later trustworthy, recurring untrustworthy behavior at 240th and 340th transactions, and (d) **Scenario 4:** Untrustworthy every 10th transaction, otherwise trustworthy. For all the scenarios, the threshold for considering a transaction as old is taken to be 150 transactions.

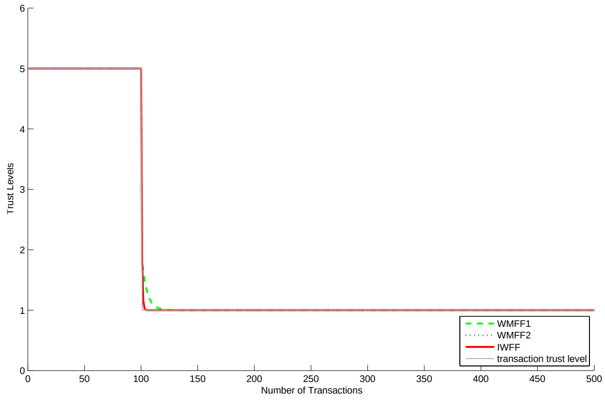

Fig. 5. Trust Evolution using WMFF1, WMFF2 and TIWFF for Scenario 2.

C. Simulation Results and Discussion

The TIWFF works on the mechanism of remembering the time of last untrustworthy transaction and reducing the difficulty of rebuilding trust gradually, as the untrustworthy transactions become older. This algorithm takes the current time, the time of last untrustworthy transaction and the time duration after which a transaction is to be considered as old, as additional inputs. In this simulation it is assumed that all the transactions are distributed uniformly in time, and so for simplification we have taken the transaction sequence number as representing the time of the transaction.

Scenario 1 tests whether the algorithm encourages the target peer to improve its behavior. As shown in Figure 4 it is found that after the initial 100 untrustworthy transactions, WMFF1 and WMFF2 make it extremely difficult for the target peer to improve its trust. Even after 400 transactions, when the target peer has performed trustworthy transactions 75% of the time, and the untrustworthy transactions have been 300 transactions old, still the target peer is not even able to achieve a trust level of 3. This is undesirable, since target peers who want to improve their behavior after untrustworthy history, will not have the motivation to perform trustworthy transactions because it will not improve their trust level significantly anyway. An effective trust evolution algorithm should nurture trust by providing incentive mechanisms and the results with WMFF1 and WMFF2 algorithms in this scenario show that they lack such an incentive mechanism.

Scenario 2 tests the agility of the trust evolution algorithm in detecting a consistent untrustworthy history. Figure 5 shows that all the three algorithms perform well for the second scenario. When an initially (100 transactions) well performing target peer starts displaying untrustworthy behavior, all the algorithms promptly respond to it and trust level is immediately brought down.

Scenario 3 tests whether an incentive mechanism of a trust evolution algorithm can be exploited by invoking the incentive mechanism condition and then performing untrustworthy transactions. The desirable behavior of a trust evolution algorithm in this scenario should be to immediately switch off the incentive mechanism and punish the target peer severely for such an attempt after being given a chance to improve from a low trust level. Figure 6 shows that WMFF1 performs poorly by not responding significantly to the untrustworthy transactions. Both WMFF1 and WMFF2 do not provide an accelerated improvement mechanism in the first place. TIWFF does provide an incentive mechanism and furthermore it is successfully able to detect an untrustworthy transaction even in the period when the algorithm was allowing the target peer an incentive mechanism to improve its trust level speedily. The untrustworthy 240th and 340th transactions are detected and the target peer is punished harshly. Instead of reaching trust level of 4 by the 250th transaction (which would have been the case if the target peer had not committed these untrustworthy transactions), it takes many more continuous trustworthy transactions for the target peer to come near trust level of 4.

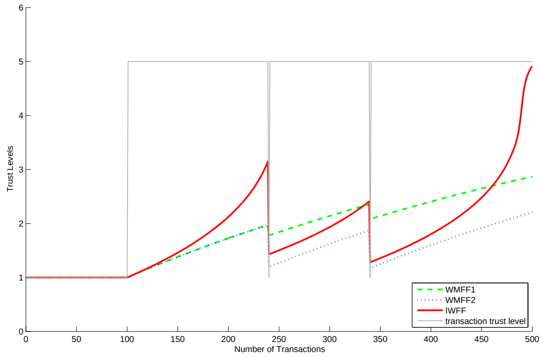

Fig. 6. Trust Evolution using WMFF1, WMFF2 and TIWFF for Scenario 3.

Scenario 4 tests whether a trust evolution algorithm allows a target peer to maintain a high trust level despite performing periodic untrustworthy transactions. An effective trust evolution algorithm should immediately detect the untrustworthy behavior and should penalize increasingly with the increasing number of cumulative untrustworthy transactions. In Figure 7 we find that WMFF2 and TIWFF are able to successfully detect the periodic untrustworthy transactions and make it increasingly difficult to achieve a high trust level after committing an untrustworthy transaction. WMFF1 displays a very bizarre trust evolution and does not consistently discourage periodic untrustworthy behavior. This deficiency in WMFF1 is a result of its discrimination between trustworthy and untrustworthy zones and we find that the trustworthiness-unaware trust evolution algorithms i.e. WMFF2 and TIWFF are more effective.

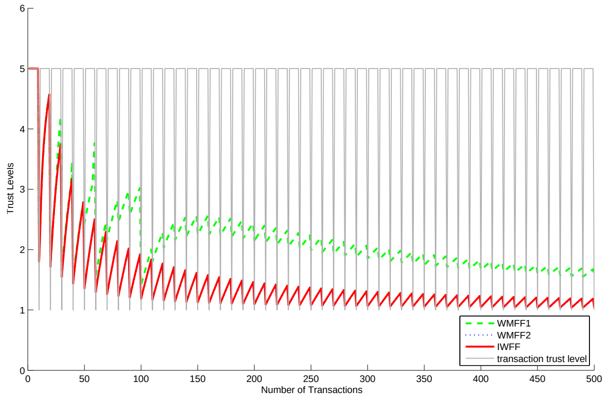

Fig. 7. Trust Evolution using WMFF1, WMFF2 and TIWFF for Scenario 4.

V. Conclusions

A number of trust evolution algorithms were tested, some taken from literature, and some developed in this work. These algorithms were evaluated against a criteria defined for effective trust evolution algorithms. It was found that TIWFF Algorithm passed all test scenarios and all evaluation criteria, performing as good as other algorithms in some scenarios, and better than other algorithms in all other scenarios. The TIWFF Algorithm is an effective trust evolution algorithm and may be used in trust models to evolve trust effectively. This algorithm introduces further improvements to the previous trust evolution algorithms. It introduces the concept of incentives to encourage peers to improve their behavior after having had an untrustworthy past.

VI. Acknowledgment

The authors wish to thank King Fahd University of Petroleum and Minerals and its Information Technology Center for the facilities utilized to perform the present work and for their support.

References

[1] D. P. Anderson, J. Cobb, E. Korpela, M. Lebofsky, and D. Werthimer, "SETI@home: An experiment in public-resource computing," *Communications of the ACM*, Vol. 45, No. 11, Nov. 2002, pp. 56–61.

[2] R. Arienghieri, E. Damiani, S. Vimercati, S. Paraboschi, and P. Samarati, "Fuzzy techniques for trust and reputation management in anonymous peer-to-peer systems," *JASIST*, Vol. 57, No. 4, 2006, pp. 528–537.

[3] F. Azzedin and M. Maheswaran, "A trust brokering system and its application to resource management in public-resource grids," *2004 International Parallel and Distributed Processing Symposium (IPDPS 2004)*, Apr. 2004.

[4] F. Azzedin, M. Maheswaran, and A. Mitra, "Trust brokering and its use for resource matchmaking in public-resource grids," *Journal of Grid Computing*, Vol. 4, No. 3, 2006, pp. 247–263.

[5] R. Brayanrd, D. Kosic, A. Rodriguez, J. Chase, and A. Vahdat, "Opus: An overlay peer utility service," *15th International Conference on Open Architectures and Network Programming (OPENARCH)*, June 2002.

[6] J. S. Chase, D. C. Anderson, P. N. Thakar, and A. M. Vahdat, "Managing energy and server resources in hosting centers," *18th Symposium Operating Systems Principles (SOSP)*, Oct. 2001.

[7] C. Corritore, B. Kracher, and S. Wiedenbeck, "On-line trust: concepts, evolving themes, a model," *International Journal of Human Computer Studies*, Vol. 58, No. 6, June 2003, pp. 737–758.

[8] B. Dragovic, S. Hand, T. Harris, E. Kotsovinos, and A. Twigg, "Managing trust and reputation in the XenoServer open platform," *1st International Conference on Trust Management (iTrust 2003)*, May 2003, pp. 59–74.

[9] I. Foster and A. Iamnitchi, "On death, taxes, and the convergence of peer-to-peer and Grid computing," *2nd International Workshop on Peer-to-Peer Systems (IPTPS03)*, Feb. 2003.

[10] T. Grandison, *Trust Management for Internet Applications*, PhD thesis, Dept. of Computing, University of London, July 2003.

[11] M. Kim and B. Noble, "Mobile networks estimation," *7th Annual Conference Mobile Computing and Networking*, July 2001.

[12] J. Lewis and A. Weigert, "Trust as a social reality," *Social Forces*, Vol. 63, No. 4, Jun. 1985, pp. 967–985.

[13] N. Luhmann, "Familiarity, confidence, trust: Problems and alternatives," in *Trust: Making and Breaking Cooperative Relations*, D. Gambetta, ed., Basil Blackwell, New York, 1988, pp. 94–107.

[14] B. Misztal, "Trust in modern societies," Polity Press, Cambridge MA, 1996.

[15] H. Nissenbaum, "Will security enhance trust online, or supplant it?," in *Trust and Distrust Within Organizations: Emerging Perspectives, Enduring Questions*, R. Kramer and K. Cook, eds., Russell Sage Publications, New York, NY, 2004, pp. 155–188.

[16] A. Oram, ed., *Peer-to-Peer: Harnessing the Power of Disruptive Technologies*, O'Reilly and Associates, Sebastopol, CA, 2001.

[17] M. A. Patton and A. Josang, "Technologies for trust in electronic commerce," *Electronic Commerce Research*, Vol. 4, No. 1, Jan. 2004.

[18] D. Quercia, S. Hailes, and L. Capra, "B-trust: Bayesian trust framework for pervasive computing," *4th International Conference on Trust Management (iTrust)*, May 2006.

[19] B. Urgaonkar, P. Shenoy, and T. Roscoe, "Resource overbooking and application profiling in shared hosting platforms," *5th Symposium on Operating Systems Design and Implementation (OSDI 2002)*, Dec. 2002.

[20] B. Yang and H. Garcia-Molina, "Designing a super-peer network," *19th International Conference on Data Engineering (ICDE)*, (Banbalore, India), Mar. 2003.