Available Online at www.ijcsmc.com

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320–088X

IJCSMC, Vol. 3, Issue. 7, July 2014, pg.183 – 188

RESEARCH ARTICLE

SORT: A SELF-ORGANIZING TRUST MODEL FOR PEER-TO-PEER SYSTEMS

Anuja Thorata
Student
M.TECH, Department of CSE,
Maheshwara Institute of Technology,
anuja27683@gmail.com

P. Prem Kumar
Assistant Professor
Department of CSE,
Maheshwara Institute of Technology,
prem1218@gmail.com

Abstract: This paper presents distributed algorithms used by a peer to reason about trustworthiness of other peers based on the available local information which includes past interactions and recommendations received from others. Peers collaborate to establish trust among each other without using a priori information or a trusted third party. A peer's trustworthiness in providing services, e.g., uploading files, and giving recommendations is evaluated in service and recommendation contexts. Three main trust metrics, reputation, service trust, and recommendation trust, are defined to precisely measure trustworthiness in these contexts. An interaction is evaluated based on three parameters: satisfaction, weight, and fading effect. When evaluating a recommendation, including to these parameters, recommender's trustworthiness and confidence about the information provided are considered. A file sharing application is simulated to understand capabilities of the proposed algorithms in mitigating attacks. For realism, peer and resource parameters are based on several empirical studies. Service and recommendation based attacks are simulated. Nine different behavior models representing individual, collaborative, and identity changing malicious peers are studied in the experiments. Observations demonstrate that malicious peers are identified by good peers. The attacks are mitigated even if they gain high reputation. Collaborative recommendation-based attacks might be successful when malicious peers make discrimination among good peers. Identity changing is not a good attack strategy.

Index Terms— Peer-to-peer systems, trust management, reputation, security

I. INTRODUCTION

P2P systems rely on collaboration of peers to accomplish tasks. Peers need to trust each other for successful operation of the system. A malicious peer can use the trust of others to gain advantage or harm. Feedbacks from peers are needed to detect malicious behavior. Since the feedbacks might be deceptive, identifying a malicious peer with high confidence is a challenge. Determining trustworthy peers requires a study of how peers can establish trust among each other. Long-term trust information about other peers can reduce the risk and uncertainty in future interactions. Interactions and feedbacks provide a means to establish trust among peers. Aberer and Despotovic developed a model that trustworthiness of a peer is measured based on complaints. A peer is assumed as trustworthy unless there are complaints about it. P-Grid provides decentralized and efficient access to trust information. Eigen trust uses transitivity of trust which allows a peer to calculate global trust values of other peers. A distributed hash table (CAN [8]) is used for efficient access to global trust information. Trust of some base peers helps to build trust among all other peers. A recommendation is evaluated according to the credibility of recommender. The experiments of Eigen trust on a file sharing application show that trust information can mitigate attacks of collaborative malicious peers. Peer Trust defines community and transaction context parameters in order to address application specific features of interactions. Four different trust calculation methods are discussed and studied in their experiments. An important aspect of trust calculation is to be adaptive for application dependent factors. We propose a Self-Organizing Trust model (SORT) that enables peers to create and manage trust relationships without using a priori information. Since preexistence of trust among peers does not distinguish a newcomer and a trustworthy one, SORT assumes that all peers are *strangers* to each other at the beginning. Peers must contribute others in order to build trust relationships. Malicious behavior quickly destroys such a relationship. Thus, Sybil attack that involves changing of pseudonym to clear bad interaction history is costly for malicious peers. In SORT, trusted peers are not needed to leverage trust establishment. A trusted peer cannot observe all interactions in a P2P system and might be a source of misleading information. A peer becomes an *acquaintance* of another peer after providing a service to it, e.g., uploading a file. Using a service from a peer is called a *service interaction*. A *recommendation* represents an acquaintance's trust information about a stranger. A peer requests recommendations only from its acquaintances. Measuring trust using numerical metrics is hard. Classifying peers as either trustworthy or untrustworthy is not sufficient. Metrics should have precision so peers can be ranked according to their trustworthiness [5], [13]. As in Eigen trust, SORT's trust metrics are normalized to take real values between 0 and 1. Eigen trust counts two peers equally trustworthy if they are assigned to the same trust value. In SORT, trust values are considered with the level of past experience. A peer with more past interactions is preferred among peers assigned to the same trust value. An interaction represents a peer's definite information about another one. A recommendation contains suspicious information. Combining these two types of information in one metric and using it to measure trustworthiness for different tasks may cause incorrect decisions. SORT defines three important trust metrics: reputation, service trust and recommendation trust. *Reputation* is the primary metric when deciding about strangers. Recommendations are used to calculate the reputation of a stranger. Providing services and giving recommendations are different tasks. A peer may be a good service provider and a bad recommender at the same time. SORT defines two contexts of trust: *service* and *recommendation contexts*. Metrics on both contexts are called *service trust* and *recommendation trust* respectively.

II. THE COMPUTATIONAL MODEL OF SORT

We make the following assumptions. Peers are indistinguishable in computational power and responsibility. There are no privileged, centralized, or trusted peers to manage trust relationships. The majority of peers are expected to be honest but some might behave maliciously. Peers occasionally leave and join the network. A peer provides services and use services of others. For simplicity of discussion, one operation is considered in the service context, e.g., file download.

A. Notations
pi denotes the *ith* peer. When *pi* uses a service of *pj* , a *service interaction* for *pi* occurs. Interactions are unidirectional. For example, if *pi* downloads a file from *pj* , no information is stored on *pj* about this download.

B. Service Trust Metric (stij)
This section describes the calculation of service trust metric. A peer first calculates competence and integrity belief values using the information about service interactions. *Competence belief* is based on how well an acquaintance satisfied the needs of interactions . *cbij* denotes the competence belief of *pi* about *pj* in the service context. Average behavior in the past interactions can be a measure of competence belief. *pi* calculates *cbij* as follows:

$$cb_{ij} = \frac{1}{\beta_{cb}} \sum_{k=1}^{sh_{ij}} (e_{ij}^k \cdot w_{ij}^k \cdot f_{ij}^k)$$

C. Reputation Metric (rij)
This section describes the calculation of the reputation metric. In the following two sections, we assume that *pj* is a stranger to *pi* and *pk* is an acquaintance of *pi*. *pi* wants to calculate *rij* value. It starts a reputation query to collect recommendations from its acquaintances. Algorithm 1 shows how *pi* selects trustworthy acquaintances and requests their recommendations. *´max* denotes the maximum number of recommendations that can be collected in a reputation query. *jj* denotes the size of a set. *pi* sets a threshold range for recommendation trust values and selects acquaintances in this range. *pi* requests recommendations from the selected peers. Then, it decreases the range and repeats the same operations. To prevent excessive network traffic, the algorithm stops when *´max* recommendations are collected or the threshold drops under *¹rt ¡ ¾rt*.

D. Further Issues
This section explains some design decisions of SORT. An extended version of this discussion can be found in. Reacting to attacks. Assume that *pi* is downloading a file from *pj*. If the file is virus infected, *pi* does special operations to protect itself against future attacks. *pi* sets satisfaction of *pj*'s past interactions to zero and creates a *permanent interaction*, which is never deleted from *pj*'s history and has zero satisfaction value. When *pi* gives a recommendation about *pj*, the receiver of the recommendation will understand that *pi* had some past experience with *pj* and then being attacked. Repeating reputation query. A peer periodically updates reputation values of its acquaintances. Knowing how an acquaintance behaves with others is helpful to understand possible threats coming from this acquaintance. Updated values may help to increase confidence on good peers and identify malicious peers in advance. Pseudonyms. A peer selects an arbitrary pseudonym and associates it with a public/private key pair so *fpseudonym, public keyg* pair becomes its identity. Peers exchange these pairs before an interaction and run a challenge-response protocol to ensure each other's identity. Thus, a malicious peer cannot use the pseudonym of another peer and take advantage of its reputation. Complaints vs. Reputation Queries. Sending complaints about a peer's malicious behavior is a way of informing other peers . However, a malicious peer may use complaints to blackmail others. Reputation queries are more resistant to blackmailing. A query may result in a misleading reputation value if the majority of recommenders maliciously collaborate to give deceptive information about the queried peer. Probability of such a collaboration is smaller than the probability that a peer individually sends a malicious complaint. Storage space. Assume that *shmax* = *rhmax* = 20 and size of a history tuple is 40 bytes. If a peer has 2000 acquaintances, 2*:*20*:*2000*:*40 = 3200*KB* is needed for both service and recommendation histories. Having 2000 acquaintances is a rare case and history size with each acquaintance will generally be less than *shmax*. Therefore, storage requirements for storing trust

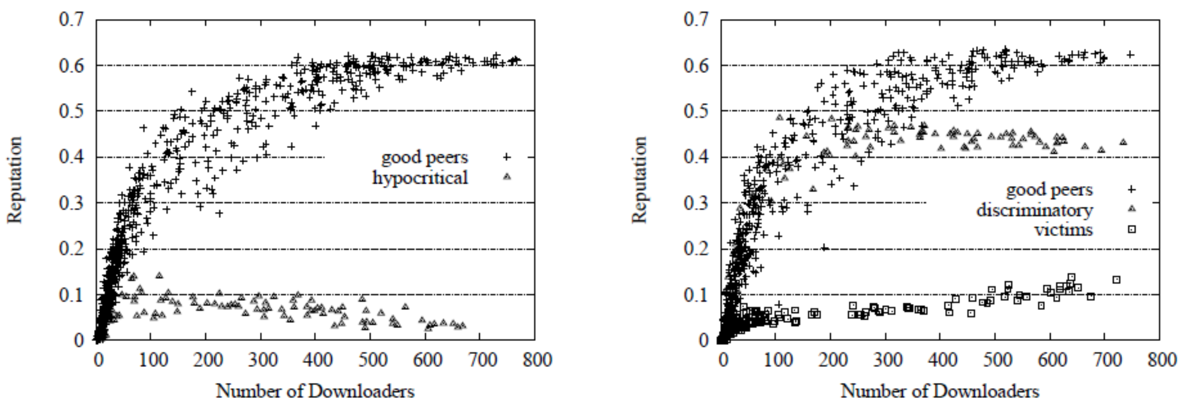
information are tolerable. Integrity checking. Although it is beyond the scope of this research, checking integrity of services needs further investigation. Downloading a file from only one uploader makes integrity checking easier since it can be done after the download finishes. When multiple uploaders are selected, checking integrity is an issue. Some uploaders may maliciously provide inauthentic content but the downloader cannot easily identify them. A naive approach might be as follows. If a peer downloads an inauthentic file, it requests hashes of the file segments from all uploaders. If there are conflicting hashes, a majority selection based on the responses of uploaders can reveal the malicious ones. Additionally, some complex methods utilizing Merkel Hashes, secure hashes, and cryptograpy can be used for online integrity checking with multiple uploaders.

III. EXPERIMENTS AND ANALYSIS

Experiments have been conducted on a file sharing application to determine the capabilities of SORT in mitigating attacks. How much recommendations are (or not) helpful in correctly identifying malicious peers, how SORT handles attacks and how much attacks can be mitigated are some questions to be studied. If a malicious peer is successful, the reason will be investigated.

A. Method

A simulation program has been implemented in Java programming language. Simulation parameters are generated based on the findings of several empirical studies vso observations about the proposed algorithms can be more realistic. Some details of the method will be explained in the next section since they are closely related with the input parameters. Downloading a file is a service interaction. A peer sharing files is called an *uploader*. A peer downloading a file is called a *downloader*. The set of peers which downloaded a file from a peer is called *downloader's* of the peer. An ongoing download/upload operation is called as a *session*. A file search request returns all online uploaders in the network. A peer downloads a file from one uploader to simplify integrity checking.



(a) Hypocritical collaborators (b) Discriminatory collaborators

IV. DISCUSSION ON FUTURE WORK

Reputation storing/collection method. Collecting trust information from acquaintances is a limiting factor of SORT. Broadcasting reputation queries is not preferred since it causes excessive network traffic. DHTs can be used to access trust information efficiently. A trust holder is assigned for each peer to store trust information. However, trust holders may behave maliciously. In SORT, a peer develops trust in its acquaintances so it can evaluate a recommendation with respect to trustworthiness of the acquaintance. In DHT approach, the information from trust holders is not reliable. A method to establish trust between a peer and trust holders need to be studied so reliability of trust holders can be evaluated. P2P system dynamics. Deletion of resources that lose popularity, addition of new peers/resources to an existing topology, multi-uploader sessions and flash crowds are some of the situations that may affect evolution of trust relationships. Studying such dynamics may help to design better trust models.

V. CONCLUSION

A self-organizing trust model for P2P networks is presented in which a peer can develop trust relations without using a priori information. Trust metrics defined on service and recommendation trust contexts help a peer to reason more precisely about capabilities of other peers in providing services and giving recommendations. If all peers are behave good, reputation of a peer is proportional to its capabilities such as network bandwidth, average online period and number of shared files. In a malicious network, service and recommendation-based attacks affect the reputation of a peer. Three individual attacker, three collaborator and three pseudo spoofer behaviors are studied. SORT mitigates service-based attacks in all scenarios. For individual attackers, hypocritical ones take more time to detect. Identification of collaborators usually takes longer than identification of an individual attacker. Pseudo spoofers are more isolated from good peers after every pseudonym change. Since good peers get more acquaintances with time, they do not prefer to interact with strangers and leave pseudospoofers isolated. Two types of collaborators present interesting behavior. Hypocritical collaborators use unfairly high recommendations and attract more good peers at the beginning. They can take advantage of SORT for their attacks. However, good peers eventually identify them and contain their attacks. Discriminatory collaborators have a better reputation than hypocritical collaborators since they do not attack 80% of the peers. However, their service-based attacks are mitigated faster since victims quickly identify them. They gain a highest recommendation trust average and cause the victims to have the lowest average. Thus, they can continue to give misleading recommendations which can be stopped if a trusted third party is used. Defining a context of trust and its related metrics increases a peer’s ability to identify and mitigate attacks in the context related tasks. Therefore, various contexts of trust can be defined to enhance security of P2P systems on specific tasks. For example, a peer might use trust metrics to select better peers when routing P2P queries, checking integrity of resources, and protecting privacy of peers.

REFERENCES

[1] P. Resnick, K. Kuwabara, R. Zeckhauser, and E. Friedman, “Reputation systems,” *Communications of ACM*, vol. 43, no. 12, pp. 45–48, 2000.

[2] C. Dellarocas, “Immunizing online reputation reporting systems against unfair ratings and discriminatory behavior,” in *Proceedings of the 2nd ACM Conference on Electronic commerce (EC)*, 2000.

[3] A. Abdul-Rahman and S. Hailes, “Supporting trust in virtual communities,” in *Proceedings of the 33rd Hawaii International Conference On System Sciences (HICSS)*, 2000.

[4] K. Aberer and Z. Despotovic, “Managing trust in a peer-2-peer information system,” in *Proceedings of the 10th International Conference on Information and knowledge management (CIKM)*, 2001. November 3, 2006 DRAFT 26

[5] S. Kamvar, M. Schlosser, and H. Garcia-Molina, “The eigentrust algorithm for reputation management in p2p networks,” in *Proceedings of the 12th World Wide Web Conference (WWW)*, 2003.

[6] L. Xiong and L. Liu, “Peertrust: Supporting reputation-based trust for peer-to-peer ecommerce communities,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 16, no. 7, pp. 843–857, 2004.

[7] K. Aberer, A. Datta, and M. Hauswirth, “P-grid: Dynamics of self-organization processes in structured p2p systems,” *Lecture Notes in Computer Science: Peer-to-Peer Systems and Applications*, vol. 3845, 2005.

[8] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker, “A scalable content addressable network,” in *Proceedings of the ACM SIGCOMM*, 2001.
[9] A. A. Selcuk, E. Uzun, and M. R. Pariente, “A reputation-based trust management system for p2p networks,” in *Proceedings of the 4th IEEE/ACM International Symposium on Cluster Computing and the Grid (CCGRID)*, 2004.
[10] J. Douceur, “The sybil attack,” in *Proceedings of the First International Workshop on Peer-to-Peer Systems (IPTPS)*, 2002.

