ABHIYANTRIKI
An International Journal of Engineering & Technology
(A Peer Reviewed & Indexed Journal)
Vol. 3, No. 4 (April, 2016) http://www.aijet.in/ eISSN: 2394-627X

Organizing Trust Model on Peer-Peer System

Nanjegowda R.*
M. Tech Student
Deptt. of Information Science & Engineering
New Horizon College of Engineering
Bangalore, India

S. Rajeshwari
Assistant Professor
Deptt. of Information Science & Engineering
New Horizon College of Engineering
Bangalore, India

Abstract

Open nature of peer-to-peer systems exposes them to malicious activity. Building trust relationships among peers can mitigate attacks of malicious peers. This paper presents trustworthiness of other peers based on past interactions and recommendations. Peers create their own trust network in their proximity by using local information available and do not try to learn global trust information. Two contexts of trust, service, and recommendation contexts are defined to measure trustworthiness in providing services and giving recommendations. Interactions and recommendations are evaluated based on importance, recentness, and peer satisfaction parameters. Additionally, recommender's trustworthiness and confidence about a recommendation are considered while evaluating recommendations. Simulation experiments on a file sharing application show that the proposed model can mitigate attacks on 16 different malicious behavior models. In the experiments, good peers were able to form trust relationships in their proximity and isolate malicious peers.

Keywords*: Trustworthiness, Reputation, Trust relationship.*

****Author for correspondence*** nanje123@gmail.com

1. Introduction

Peer-to-peer systems rely on collaboration of peer to accomplish task. Ease of performing the malicious activity is a threat for the security of P2P computer system. Creating long term trust relationship among peers can provide a secure environment by reducing risk and uncertainty in future P2P interaction, however establishing trust in an unknown entity is difficult in such a malicious environment furthermore, trust is a social concept and hard to measure with numerical value. Metrics are needed to represent trust in computational models, classifying peers as either of trustworthy or the untrustworthy is not sufficient in most cases. Metrics should have precision so peers can be ranked according to trustworthiness. Interactions and the feedbacks of peers provide the information to measure trust among peers. Interactions with a peer provide the certain information about the peer but feedbacks might contain deceptive information.

2. Literature Survey

[1] Trust management: Managing trust is a problem of particular importance in peer-to-peer environments where one frequently encounters unknown agents. The method can be

implemented in a fully decentralized P2P environment. Scale well for large number of participants. It is not robust to malicious collective peers.
[2] Eigen trust: Eigen trust algorithm for reputation management in P2P network. All peers in the network cooperate to compute and store the global trust vector. Each peer stores and computes its' own global trust value. It cannot distinguish between new comers and malicious peers. Difficult to calculate reputation values when peers join and leave.
[3] Reputation: Reputation-based trust management protocol for P2P networks where users rate the reliability of parties they deal with, and share this information with their peers. The protocol helps establishing trust among good peers as well as identifying the malicious ones. Digital signature is used for authentication. Update of the credibility ratings is slightly more complex.
[4] Computational: This concerns the relationships based on trust and reputation. Agents can obtain data from other agents. Not all kinds of environment are suitable to apply these mechanisms.
[5] Peer trust: Here we are building trust model from one peer about onther peer. Exploring mechanisms to make peer trust model more robust against malicious behavior such as collusion among peers. The five factors used in their trust model must be retrieved with a heavy overhead.
[6] Trust model: To build a general trust metric that provides an effective measure for capturing the trustworthiness of peers, addresses the fake or misleading feedbacks, and has the capability to adapt to different communities and situations. The increasing complexity of large distributed system such as the Internet can be managed more effectively. This approach is that every agent must keep rather complex and very large data structure represents a kind of global knowledge about the whole network.
[7] Reputation certificate: Communication cost is low. Protect the integrity of the reputation RCert is used. It cannot prevent malicious participants collude to distort the reputation information.
[8] Service rating: Service ratings are normalized values ranging from -1.0 to 1.0 with 0 indicating a neutral rating. It improves accuracy by removing the assumption of correlation between service quality and feedback quality. Do not explicitly define how reputations and records of ratings are stored.

3. Problem Statement

Existing system of an authority, a central server is a preferred way to store and manage trust information, e.g., eBay. The central server securely stores trust information and defines trust metrics. Since there is no central server in most P2P systems, peers organize themselves to store and manage trust information about each other. Management of trust information is dependent to the structure of P2P network. In distributed hash table (DHT) – based approaches, each peer becomes a trust holder by storing feedbacks about other peers. Global trust information stored by trust holders can be accessed through DHT efficiently. In unstructured networks, each peers stores trust information about peers in its neighborhood or peers interacted in the past. A peer sends trust queries to learn trust information of other peers. A trust query is either flooded to the network or sent to neighborhood of the query initiator.

Disadvantages of Existing System

- Calculated trust information is not global and does not reflect opinions of all peers.
- Classifying peers as either trustworthy or untrustworthy is not sufficient in most cases. Metrics should have precision so peers can be ranked according to trustworthiness.

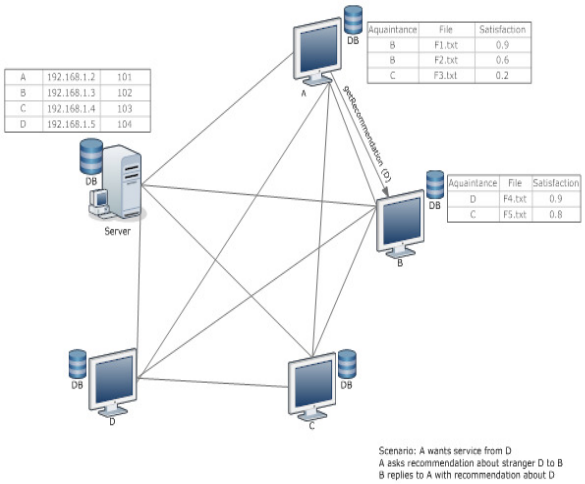
- Trust models on P2P systems have extra challenges comparing to e-commerce platforms. Malicious peers have more attack opportunities in P2P trust models due to lack of a central authority.

Solution Strategy

Proposed system introduced Self-Organizing Trust model (SORT) that aims to decrease malicious activity in a P2P system by establishing trust relations among peers in their proximity. No a priori information or a trusted peer is used to leverage trust establishment. Peers do not try to collect trust information from all peers. Each peer develops its own local view of trust about the peers interacted in the past. In this way, good peers form dynamic trust groups in their proximity and can isolate malicious peers. Since peers generally tend to interact with a small set of peers forming trust relations in proximity of peers helps to mitigate attacks in a P2P system.

4. System Architecture of SORT

This is a hybrid peer to peer approach. Here server is used only to store the location of the files. Each and every system contains the database. It contains three parameters i.e. acquaintances, file name and satisfaction parameters. Each peer develops its own local view of trust about the peers interacted in the past. In this way, good peers form dynamic trust groups in their proximity and can isolate malicious peers.



* All nodes will store the trust information related to every other node in the network.
* Network traffic will be heavy.
* Classifying peers as either trustworthy or untrustworthy is not sufficient in most cases.
* Figure
* Users and their characteristics
* We have three users who use the system and execute our project successfully. These users make it possible to show the success of the proposed concept in a systematic way.
* Administrator
  - Create topology.
  - Add new node to the topology.
  - Provide a specification for each node.
  - Set the path between the nodes and give the path cost.
* Sender
  - Can send any type of file or text for transmission.
  - Store satisfaction values for each transmission.
  - Can check for reputation of any acquaintance from local history.
    If local history does not contain any satisfaction record for any node then if may check for reputation from other peers of service provider.

* Receiver
  Receives the text message or receives the files which are requested from the service provider.

* Central Server
  Central server is a preferred way to store and manage trust information, e.g., eBay. The central server securely stores trust information and defines trust metrics. Since there is no central server in most P2P systems, peers organize themselves to store and manage trust information about each other. Management of trust information is dependent to the structure of P2P network. In distributed hash table (DHT) - based approaches, each peer becomes a trust holder by storing feedbacks about other peers. Global trust information stored by trust holders can be accessed through DHT efficiently. In unstructured networks, each peers stores trust information about peers in its neighborhood or peers interacted in the past. A peer sends trust queries to learn trust information of other peers. A trust query is either flooded to the network or sent to neighborhoods of the query initiator.

* Disadvantages
  - Calculated trust information is not global and does not reflect opinions of all peers. Classifying peers as either trustworthy or untrustworthy is not sufficient in most cases. Metrics should have precision so peers can be ranked according to trustworthiness.
  - Trust models on P2P systems have extra challenges comparing to e-commerce platforms. Malicious peers have more attack opportunities in P2P trust models due to lack of central authority. Five common attacks in P2P trust models: self-promoting, white-washing, slandering, orchestrated, and denial of service attacks.

5. Trust Model

We organize SORT that aims to decrease malicious activity in a P2P system by establishing trust relations among peers in their proximity. No a priori information or a trusted peer is used to leverage trust establishment. Peers do not try to collect trust information from all peers. Each peer develops its own local view of trust about the peers interacted in the past. In this way, good peers form dynamic trust groups in their proximity and can isolate malicious peers. Since peers generally tend to interact with a small set of peers forming trust relations in proximity of peers helps to mitigate attacks in a P2P system.

Advantage

Recommendation based attacks were contained except when malicious peers are in large numbers, e.g., 50% of all peers. Experiments on SORT show that good peers can defend themselves against malicious peers metrics let a peer assess trustworthiness of other peers based on local information. Service and recommendation contexts enable better measurement of trustworthiness in providing services and giving recommendations.

a) Preliminary Notations

The pi denotes i^{th} peer. When pi uses a service of another peer, it is an interaction for pi. Interactions are unidirectional. For example, if pi downloads a file from pj, it is an interaction for pi and no information is stored on pj. If pi had at least one interaction with pj, pj is an acquaintance of pi. Otherwise, pj is a stranger to pi. Ai denotes pi's set of acquaintances. A peer stores a separate history of interactions for each acquaintance. Shij denotes pi's service history

with pj where shij denotes the current size of the history. Shmax denotes the upper bound for service history size. Since new interactions are appended to the history, SHij is a time ordered list. After finishing an interaction, pi evaluates quality of service and assigns a satisfaction value for the interaction. Let 0 _ sk _ 1 denote pi's satisfaction about k^{th} interaction with pj. If an interaction is not completed: skij ¼ 0. An interaction's importance is measured with a weight value. Let 0 _ wkij _ 1 denote the weight of k^{th} interaction of pi with pj.

b) Computational Model Sort

We make the following assumptions. Peers are equal in computational power and responsibility. There are no centralized or trusted peers to manage trust relationships. Peers occasionally leave and join the network. A peer provides services and uses services of others. For simplicity of discussion, one type of interaction is considered in the service context, i.e. file download. A file sharing simulation program is implemented in Java to observe results of using SORT in a P2P environment. Some questions studied in the experiments are as follows: how SORT handles attacks, how much attacks can be mitigated, how much recommendations are (not) helpful in correctly identifying malicious peers, and what type of attackers are the harmful.

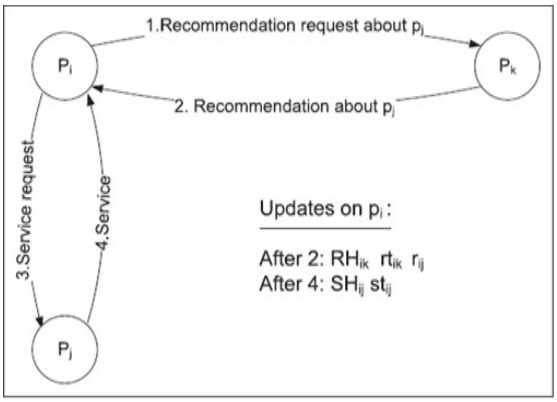


Fig.: Operation when receiving a recommendation and having an interaction

6. Methodology

A trust model for P2P networks is presented, in which a peer can develop a trust network in its proximity. A peer can isolate malicious peers around itself as it develops trust relationships with good peers. Two context of trust, service and recommendation contexts are defined to measure capabilities of peers in providing services and giving recommendations. Interactions and recommendations are considered with satisfaction. A recommendation contains the recommender's own experience, information from its acquaintances, and level of confidence in the recommendation. These parameters provided us a better assessment of trustworthiness.

a) Analysis on Individual Attacks

This section explains the results of experiments on individual attackers. For each type of individual attacker, two separate network topologies are created: one with 10 percent malicious and one with 50 percent malicious. Each network topology is tested with four trust calculation methods. In the experiments, a hypocritical attacker behaves malicious in 20 percent of all interactions. A discriminatory attacker selects 10 percent of all peers as victims. An oscillatory attacker behaves well for 1,000 cycles and malicious for 100 cycles.

Service-based attacks when a malicious peer uploads an infected/inauthentic file, it is recorded as a service-based attack. Number of attacks in No Trust method is considered as the base case to understand how many attacks can happen without using trust information. Then, number of attacks observed for each trust calculation method is compared with the base case to determine the percentage of attacks prevented. In the table, NoRQ and Flood RQ denote "no reputation query" and "Flood reputation query" methods.

Naive Collaborators always upload infected/inauthentic files to good peers and give unfairly lower commendations about them. Discriminatory Collaborators select a group of peers as victims. They upload infected/ inauthentic files to the victims and give unfairly low recommendations bout them. They upload authentic files to non victim peers. They also give fair recommendations about non victim peers. Hypocritical Collaborators upload infected or in authentic files to good peers or give unfairly lower commendations about them with x percent probability. In the other cases, they behave as good peers.

b) Analysis on Individual Pseudospoofers

This section explains the results of experiments on individual pseudo spoofs. Pseudo spoofs change their pseudonyms after every 1,000 cycles. The values obtained by comparing the base case with each trust calculation method. After every pseudonym change, attackers become strangers to others. This behavior has two effects:

- Pseudospoofers clear their bad history. Hence a good peer may interact with them when it cannot find more reliable uploaded, which increases attacks.
- Pseudospoofer become more isolated from good peers. They lose their ability to attract good peers with time, which decreases attacks.

In all experiments, No RQ method performs 20-40 percent worse than other trust calculation methods. Since no recommendation is collected in No RQ method, chance of selecting an attacker again is higher after every pseudonym change. Therefore, SORT and FloodRQ methods have better results in the experiments. Recommendations increase the chance of finding good peers among strangers.

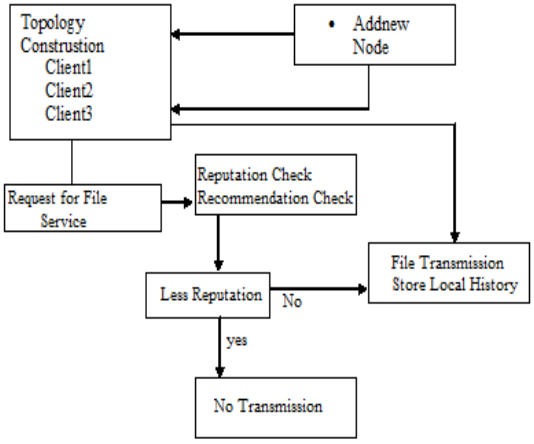


Fig: Layout of the System

In SORT, a peer interacts less with strangers as its set of acquaintances grows. Therefore, rate of service-based attacks decreases with time. In all cases, SORT's prevention ratio for service-based attacks is close to Flood RQ method. However, Flood RQ method causes 7-10 times more recommendation traffic than SORT. The difference in misleading recommendations is much higher as explained below. Thus, SORT has a better performance trade off than Flood RQ method. Recommendation based attacks: In the simulations, when a malicious peer gives a misleading recommendation, it is recorded.

7. User View of Product Use Approach

a) Topology Creation Module

Given Input: Entering the number of nodes and node information like node name, port number, system name that needs to be constructed.

Expected Output: Topology gets constructed.

b) Adding New Node to Topology Module

Given Input: The node which needed to be add to the existing topology.

Expected Output: Node successfully added to the Topology.
The information can be see in the Database.

c) File Upload Module
Given Input: Any type of file can be chosen.
Expected Output: File will be uploaded to node files folder.

d) File Selection Module
Given Input: File should be selected from available files.
Expected Output: Request should go to the server for node address containing file and path through which file will be received.

e) File Transmission Module
Given Input: Click receive button after selecting file.
Expected Output: Reputation and recommendation should be checked before receiving file from service provider and then file should be received.

8. Conclusions and Further Work

A trust model for P2P networks is presented, in which a peer can develop a trust network in its proximity. A peer can isolate malicious peers around itself as it develops trust relationships with good peers. Two context of trust, service and recommendation contexts are defined to measure capabilities of peers in providing services and giving recommendations. Interactions and recommendations are considered with satisfaction, weight and fading effect parameters. A recommendation contains the recommender's own experience, information from its acquaintances and level of confidence in the recommendation. These parameters provided us a better assessment of trustworthiness. Individual, collaborative, and pseudonym changing attackers are studied in the experiments. Damage of collaboration and pseudo spoofing is dependent to attack behavior. Although recommendations are important in hypocritical and oscillatory attackers, pseudospoofers and collaborators; they are less useful in naive and discriminatory attackers. SORT mitigated both service and recommendation based attacks in most experiments. However, in extremely malicious environments such as a 50 percent malicious network, collaborators can continue to disseminate large amount of misleading recommendations.

References

[1] AB Can, & B. Bhargava. (2013). SORT: A Self-ORganizing Trust Model for Peer-to-Peer Systems. *IEEE Transactions on Dependable and Secure Computing*. 10(1).
[2] R Zhou, & K Hwang. (2007). Power trust: A Robust and Scalable Reputation System for Trusted Peer-to-Peer Computing. *IEEE Trans. Parallel and Distributed Systems*. 18(4), pp. 460-473.
[3] F Cornelli, E Damiani, SDC di Vimercati, S Paraboschi, & P Samarati. (2002). Implementing a Reputation-Aware Gnutella Servent. *Proc. Networking 2002 Workshops Web Eng. and Peer-to-Peer Computing*.
[4] B Yu, MP Singh, & K Sycara. (2004). Developing Trust in Large-Scale Peer-to-Peer Systems. *Proc. IEEE First Symp. Multi-Agent Security and Survivability*.
[5] S Boyd, A Ghosh, B Prabhakar, & D Shah. (2006). Randomized Gossip Algorithms. *IEEE/ACM Trans. Networking*. 52(6), pp. 2508-2530.