International Journal of Communication Networks and Information Security (IJCNIS) Vol. 4, No. 1, April 2012

A Secure and User Privacy-Preserving Searching Protocol for Peer-to-Peer Networks

Jaydip Sen

Innovation Lab, Tata Consultancy Services Ltd.
[2]Bengal Intelligent Park, Salt Lake Electronic Complex, Kolkata 700091, INDIA,
jaydip.sen@acm.org

***Abstract*:** File sharing peer-to-peer networks have become quite popular of late as a new paradigm for information exchange among large number of users in the Internet. However, these networks suffer from several problems such as fake content distribution, free riding, whitewashing, poor search scalability, lack of a robust trust model and absence of user privacy protection mechanism. In this paper, a secure and efficient searching scheme for peer-to-peer networks has been proposed that utilizes topology adaptation by constructing an overlay of trusted peers where the neighbors are selected based on their trust ratings and content similarities. While increasing the search efficiency by intelligently exploiting the formation of semantic community structures among the trustworthy peers, the scheme provides a highly reliable module for protecting the privacy of the users and data in the network. Simulation results have demonstrated that the proposed scheme provides efficient searching to good peers while penalizing the malicious peers by increasing their search times.

***Keywords*:** P2P networks, topology adaptation, trust, reputation, semantic community, malicious peer, user privacy.

1. Introduction

The term *peer-to-peer* (P2P) system encompasses a broad set of distributed applications which allow sharing of computer resources by direct exchange between systems. The goal of a P2P system is to aggregate resources available at the edge of Internet and to share it cooperatively among users. Specially, the file sharing P2P systems have become popular as a new paradigm for information exchange among large number of users in the Internet. They are more robust, scalable, fault tolerant and offer better availability of resources than the traditional client server model. Depending on the presence of a central server, P2P systems can be classified as centralized or decentralized [11]. In decentralized architecture, both resource discovery and resource download are distributed. Decentralized P2P architectures may further be classified as structured or unstructured networks. In structured networks, there are certain restrictions on the placement of contents and the network topologies. In unstructured P2P networks, however, placement of contents is unrelated to the topologies of the network. Unstructured P2P networks perform better than their structured counterparts in dynamic environments. However, they need efficient search mechanisms and they also suffer from numerous problems such as: fake content distribution, free riding (peers who do not share, but consume resources), whitewashing (peers who leave and rejoin the system in order to avoid penalties) and lack of scalability in searching. Open and anonymous nature of P2P applications lead to complete lack of accountability of the contents that a peer may put in the network. The malicious peers often use these networks to do content poisoning and to distribute harmful programs such as Trojan Horses and viruses [12]. *Distributed reputation based trust management systems* have been proposed by researchers to provide protection against malicious content distribution [1]. The main drawbacks of these schemes are their high message exchange overheads and their susceptibility to misrepresentation. Guo et al. have proposed *trust-aware adaptive P2P topology* to control free-riders and malicious peers [7]. In [3] and [25], topology adaptation is used to reduce inauthentic file downloads. However, these schemes do not work well in unstructured networks. Poor search scalability is another problem. Traditional mechanisms such as controlled flooding, random walker and topology evolution all lack scalability. Zhuge et al. have proposed trust-based probabilistic search algorithm called *P-walk* to improve search efficiency and to reduce unnecessary traffic in P2P networks [28]. In P-walk, neighboring peers assign trust scores to each other. During routing, peers preferentially forward queries to the highly ranked neighbors. However, its performance in large-scale unstructured network is questionable. To combat free riders, various trust-based incentive mechanisms are presented in [26]. Most of these mechanisms, however, involve large overhead of computations.

To combat the problem of inauthentic downloads as well as to improve search scalability while protecting the privacy of the users, this paper proposes an adaptive trust-aware algorithm that is robust and scalable. This work is an extension of our previously published scheme which is based on construction of an overlay of trusted peers where neighbours are selected based on their trust ratings and content similarities [14, 15]. It increases search efficiency by taking advantage of implicit semantic community structures formed as a result of topology adaptation since most of the queries are resolved within the community [14, 15]. However, the novel contribution of the current work is that it combines the functionalities of a robust trust management model and the semantic community formation to provide a secure and efficient searching scheme while protecting the privacy of the users. The trust management scheme segregates the honest peers from malicious peers, based on both first-hand and second-hand information, and the semantic community formation allows topology adaptation to form cluster of peers sharing similar contents in the network. The formation of the semantic communities also enables the scheme to form a neighborhood of trust which is utilized to protect user privacy in the network.

The rest of the paper is organized as follows. Section 2 discusses some related work. Section 3 presents the proposed algorithm for secure and privacy-aware searching. For the

benefit of the readers, we present the entire algorithm including the one presented in [14, 15]. As mentioned in the previous paragraph, the scheme presented in this paper has a robust trust management model and privacy preserving module which were not present in the scheme described in [14, 15]. Section 4 first introduces various metrics to measure the performance of the proposed algorithm, and then presents the simulation results. A brief discussion is also made on the comparative analysis of the performance of the proposed scheme with some of the existing similar schemes in the literature. Section 5 concludes the paper while highlighting some future scope of work.

2. Related Work

In [5], a searching mechanism is proposed that is based on discovery of trust paths among the peers in a peer-to-peer network. A global trust model based on distance-weighted recommendations has been proposed in [10] to quantify and evaluate the peers in a peer-to-peer network. In [3], a protocol named *adaptive peer-to-peer technologies* (APT) for the formation of adaptive topologies has been proposed to reduce spurious file download and free riding, where a peer connects to those peers from whom it is most likely to download satisfactory content. It adds or removes neighbors based on *local trust* and *connection trust* which are decided by its transaction history. The scheme follows a defensive strategy for punishment where a peer equally punishes both malicious peers as well as neighbors through which it receives response from malicious peers. This strategy is relaxed in the *reciprocal capacity-based adaptive topology protocol* (RC-ATP), where a peer connects to others which have higher reciprocal capacity [25]. Reciprocal capacity is defined based on the peer's capacity of providing good files and of recommending source of download. While RC-ATP provides better network connectivity than APT, and reduces the cost of inauthentic downloads, it has a large overhead of topology adaptation. In [9], an algorithm has been presented to reduce the number of downloads of inauthentic files in a file-sharing peer-to-peer network. Each peer is assigned a unique global trust value that is computed based on the historical activities of the peer in the network. A distributed and secure method based on Power iteration is also presented for computing the global trust values of the peers. Based on the global trust values, the malicious peers are identified and isolated from the network. Xio et al. have proposed an *adaptive connection establishment* (ACE) protocol that constructs an overlay multicast tree by including each source peer and the neighboring peers within a certain diameter from the source peer [27]. It further optimizes the connecting edges in the overlay graph that are not included in the tree while retaining the scope of the search. The protocol is completely distributed since the peers do not need global knowledge of the whole overlay network while using the search protocol.

There are some significant difference between the proposed algorithm and APT and RC-ATP. First, in the proposed scheme, the links in the original overlays are never deleted to avoid network partitioning. Second, the robustness of the proposed protocol in presence of malicious peers is higher than that of APT and RC-ATP protocols as shown in the experimental results. Third, as APT and RC-ATP both use flooding to locate resource, they have poor search scalability. The proposed scheme takes the advantages of semantic communities to improve QoS of search. Fourth, APT and RC-ATP do not employ any robust trust model for security in searching and for user identity and data privacy protection. The central part of the proposed searching mechanism in this paper is a robust trust management model. Finally, unlike APT and RC-ATP, the proposed algorithm scheme punishes malicious peers by blocking query initiated by them. This ensures that malicious peers are not allowed to consume the resources and services available in the network.

3. The Secure and Privacy-Aware Searching Algorithm

This section is divided into two sub-sections. In Section 3.1, the various parameters and network environment of p2p networks are discussed. In Section 3.2, the proposed search algorithm is presented with the details of its features related to security and user privacy protection. Both these sub-sections are divided into several sub-sections, each one discussing a particular feature of the proposition.

3.1 The network environment

To derive meaningful conclusion from the proposed algorithm, the proposed scheme have been modelled in P2P networks in a realistic fashion. The factors that are taken into consideration for design of the scheme are as follows.

3.1.1 Network topology

The topology of a P2P network plays an important role for the analysis of trust management among its peers and for designing a searching scheme. Following the work in [3] and [25], the network has been modelled as a *power law graph*. In a power law network, degree distribution of nodes follows power law distribution, i.e. fraction of nodes having degree L is L^{-k} where k is a network dependent constant. Prior to each simulation cycle, a fixed fraction of peers chosen randomly is marked as malicious. As the algorithm executes, the peers adjust topology locally to connect to the peers which have better chance to provide good files in future, and drop malicious peers from their neighborhood. The network links are categorized into two types: *connectivity link* and *community link*. The connectivity links are the edges of the original power law network which provide seamless connectivity among the peers. To prevent the network from being partitioned, these links are never deleted. On the other hand, community links are added probabilistically between the peers who know each other, and have already interacted with each other before. A community link may be deleted when the perceived trustworthiness of a peer falls in the perception of its neighbors. A limit is put on the additional number of edges that a node can acquire to control bandwidth usage and query processing overhead in the network. This increase in network load is measured relative to the initial network degree (corresponding to connectivity edges). Let *final_degree(x)* and *initial_degree(x)* be the initial and final degree of a node x. The *relative increase in connectivity* (RIC) as computed in (1) is constrained by a parameter called *edge limit*.

$$RIC(x) = \frac{final\_degree(x)}{initial\_degree(x)} \leq edge\_limit \quad (1)$$

3.1.2 Content distribution

The dynamics of a P2P network are highly dependent on the volume and variety of files each peer chooses to share. Hence a model reflecting real-world P2P networks is required. It has been observed that peers are in general interested in a subset of the contents in the P2P network [4]. Also, the peers are often interested only in the files from a few content categories. Among these categories some are more popular than others. It has been shown that Gnutella content distribution follows *zipf distribution* [13]. Keeping this in mind, both content categories and file popularity within each category is modelled with *zipf distribution* with $\alpha = 0.8$.

Content distribution model: The content distribution model in [13] is followed for the purpose of simulation. In this model, each distinct file $f_{c,r}$ is abstractly represented by the tuple (c, r), where c represents the content category to which the file belongs, and r represents its popularity rank within a content category c. Let content categories be $C = \{C_1, C_2,....,C_{32}\}$. Each content category is characterized by its popularity rank. For example, if $C_1 = 1$, $C_2 = 2$ and $C_3 = 3$, then C_1 is more popular than C_2 and hence it is more replicated than C_2 and so on. Also there are more files in category C_1 than C_2.

Table 1. Content distribution in peers

Peers	Content categories
P1	C1, C2, C3
P2	C3, C4, C6, C7
P3	C2, C4, C7, C8
P4	C1, C2
P5	C1, C5, C6

Each peer randomly chooses between three to six content categories to share files and shares more files belonging to more popular categories. Table 1 shows an illustrative content distribution among 5 peers. The category C_1 is more replicated as it is the most popular category. Peer 1 (P_1) shares files in three categories: C_1, C_2, C_3, where it shares maximum number of files in category C_1, followed by category C_2 and so on. On the other hand, Peer 3 (P_3) shares maximum number of files in category C_2 as it is the most popular among the categories chosen by it.

3.1.3 Query initiation model

The authors in [13] suggest that peers usually query for files which are available in the network, and are in the content category of their interest. In each cycle of simulation, active peers issue queries. However, the number of queries a peer issues may vary from peer to peer. Using the *Poisson* distribution this is modelled as follows. If M is the total number of queries to be issued in each cycle of simulation, and N is the number of peers present in the network, query rate $\lambda = \frac{M}{N}$ is the mean of the Poisson process. The probability that a peer issues K queries in a cycle is: $p(\#ofqueries = K) = \frac{e^{-K}\lambda^K}{K!}$. The probability that a peer issues query for the file r depends on the peer's interest level in category c and rank r of the file within that category.

3.1.4 Trust management engine

Trust and reputation based models are widely used for security systems in self-organizing networks like mobile ad hoc networks (MANETs), wireless sensor networks (WSNs), and wireless mesh networks (WMNs) [16, 17, 18, 19, 20]. These models can be exploited in unstructured and dynamic p2p networks as well. The make the proposed searching algorithm secure, a trust management engine is designed which helps the peers to compute trust ratings of other peers from past transactions as well as recommendation from its neighbor. For computation of trust values for the peers, a method similar to the one proposed in [6] is followed. The framework employs a *beta distribution* for reputation representation, updates and integration. The first-hand information and second-hand (recommendation from neighbors) are combined to compute the reputation value of a peer. The weight assigned by a peer i to a second-hand information received from a node k is a function of reputation of node k as maintained in node i. For each peer j, a reputation R_{ij} is computed by a neighbor peer i. The reputation is embodied in the *Beta model* which has two parameters: α_{ij} and β_{ij}. While α_{ij} represents the number of successful transactions (i.e. authentic file downloads) that peer i had with peer j, β_{ij} represents the number of unsuccessful transactions (i.e., unauthentic file downloads). The reputation of peer j as maintained by peer i is computed using (2).

$$R_{ij} = Beta(\alpha_{ij} + 1, \beta_{ij} + 1) \quad (2)$$

The trust metric of a peer is the expected value of its reputation and is given by (3).

$$T_{ij} = E(R_{ij}) = E(Beta(\alpha_{ij} + 1, \beta_{ij} + 1)) = \frac{\alpha_{ij} + 1}{\alpha_{ij} + \beta_{ij} + 2} \quad (3)$$

The second-hand information is presented to peer i by its neighbor peer k. The peer i receives the reputation R_{kj} of peer j from peer k in the form of the two parameters α_{kj} and β_{kj}. After receiving this new information, the peer i combines it with its current assessment R_{ij} to obtain a new reputation R_{ij}^{new} as shown in (4).

$$R_{ij}^{new} = Beta(\alpha_{ij}^{new}, \beta_{ij}^{new}) \quad (4)$$

In (4), the values of α_{ij}^{new} and β_{ij}^{new} are given by (5) and (6) as follows.

$$\alpha_{ij}^{new} = \alpha_{ij} + \frac{2\alpha_{ik}\alpha_{kj}}{(\beta_{ik} + 2)(\alpha_{kj} + 2)(\alpha_{kj} + \beta_{kj} + 2)(2\alpha_{ik})} \quad (5)$$

$$\beta_{ij}^{new} = \beta_{ij} + \frac{2\alpha_{ik}\beta_{kj}}{(\beta_{ik} + 2)(\alpha_{kj} + \beta_{kj} + 2)(2\alpha_{ik})} \quad (6)$$

The proposed trust model gives more weight to recent observations, which is used for updating the reputation value using direct observation. For updating the reputation value using the second-hand information, *Dempster-Shafer theory* [24] and the *belief discounting model* [8] are used. The reputation of a recommending peer is automatically taken into account while computing the reputation of the reported peer. This eliminates the need of a separate deviation test. As mentioned earlier in this sub-section (i.e., Section 3.1.4), the trust value of a peer is computed as the statistical expected value of its reputation. The trust value of a peer lies in the interval [0, 1]. Peer i considers peer j as trustworthy if $S_{ij} \geq 0.5$, and malicious if $S_{ij} < 0.5$.

3.1.5 Identity of the peers

Each peer generates a 1024 bit public/private RSA key pair. The public key serves as the identity of the peer. The identities are persistent and they enable two peers that have exchanged keys to locate and connect to one another whenever the peers are online. In addition, a *distributed hash table* (DHT) is maintained that lists the transient IP addresses and port numbers for all peers for all applications running of the peers. DHT entries for the peer i are signed by i and encrypted using its public key. Each entry is indexed by a 20 byte randomly generated shared secret, which is agreed upon during the first successful connection between the two peers. Each peer's location in the DHT is independent of its identity and is determined by hashing the client's current IP address and DHT port. This inhibits systematic monitoring of targeted regions of the DHT key space since the region for which each peer is responsible is determined by that peer's network address and port.

3.1.6 Node churning model

In P2P networks, a large number of peers may join and leave at any time. This activity is termed as *node churning*. To simulate node churning, prior to each *generation* (a set of consecutive searches), a fixed percentage of nodes are chosen randomly as inactive. These peers neither initiate nor respond to a query in that generation and join the system latter with their LRU structure cleared. Since in a real world network, even in presence of churning, the approximate distribution of content categories and files remain unchanged, the contents of peers undergoing churn are exchanged and new contents are assigned to the peers thereby keeping the content distribution unchanged.

3.1.7 Threat model

Malicious peers adopt various strategies (threat models) to conceal their behavior and disrupt activities in the network. Two threat models are considered in the proposed scheme. The peers who share good quality files enjoy better topological due to topology adaptation. In threat model *A*, malicious peers attempt to circumvent this by providing good file occasionally with probability, known as *degree of deception* to lure other peers to form communities with them. In threat model *B*, a group of malicious peer joins the system and provides good files until their connectivity reaches a maximum value, i.e., the *edge limit*. The peers then start acting maliciously by spreading fake contents in the network.

3.2 The proposed searching algorithm

The network learns trust information through the search and updates trust information and adapts topology based on the outcome of the search. The following criteria are kept in mind while designing the algorithm: (a) It should improve search efficiency as well as search quality (i.e., authentic file downloads). (b) It should have minimal overhead in terms of computation, storage and message passing. (c) It should provide incentives to the peers which share large number of authentic files. (d) It should be self-policing in the sense that a peer should be able adjust its search strategy based on the local estimate of network connectivity. (e) It should be able to protect the privacy of the users. The major steps of execution of the algorithm are: (i) search, (ii) topology adaptation, (iii) trust computing and verification, and (iv) user privacy protection. Each of these steps is discussed in the following.

3.2.1 Searching scheme

A *time to live* (TTL) bound search is used. At each peer, the query is forwarded to a subset of neighbors; the number of neighbors is decided based on the local estimate connectivity. The *connectivity index* for peer x is denoted as $Prob_{com}(x)$ and is given by (7).

$$Prob_{com}(x) = \frac{degree(x) - initial\_degree(x)}{initial\_degree(x)(edge\_limit - 1)} \quad (7)$$

When $Prob_{com}$ for a node is low, the peer has the capacity to accept new community edges and expand community structures. Higher the value of $Prob_{com}$, it is less likely that the neighbors will disseminate the queries. As the algorithm executes the connectivity of good nodes increases and reaches a maximum value. At this time, the peers focus on directing the queries to appropriate communities which may host the specific file rather than expanding communities. For example, if peer i can contact at most 10 neighbors and $Prob_{com}$ of j is 0.6, it forwards query to: 10 x (1 - 0.6) = 4 neighbors only. The search strategy is changed from initial TTL limited BFS to directed DFS with the restructuring of the network. The search process has two steps: *query initiation* and *query forward* as explained in the following.

3.2.1.1 Query initiation

The initiating peer forms a query packet containing the name of the file (c, r) and forwards it to some of its neighbors along with $Prob_{com}$ and TTL values. The query is disseminated using the following *neighbor selection rule*. The neighbors are ranked based on both their trustworthiness and the similarity of interests. Preference is given to the trusted neighbors sharing similar contents. Among the trusted neighbors, community members having their contents matched to the query are preferred. If the number of community links is not adequate enough, the query is forwarded through connectivity links also. The various cases of neighbor selection are illustrated in Fig. 1.

It is assumed that in each case only two neighbors are selected. When the query (c_2, f_4) reaches the peer P, following four cases may occur. In Case 1, the peer P has

sufficient number of community neighbors sharing file in the category c_2. Hence, these peers are chosen for forwarding the query. In Case 2, the number of community neighbors sharing the requested category of file is not sufficient enough. In this scenario, the community neighbors sharing c_2 and c_6 categories of files are preferred over the connectivity neighbor sharing file category c_2 for forwarding the query. In Case 3, there is only one community neighbor that shares file category c_2. Hence that neighbor is chosen for the purpose of query forwarding. Among the remaining connectivity neighbors, the most trusted one containing $c6$ is selected. In Case 4, there are no community neighbors. Assuming that peer P has the same level of trust for all its neighbors, the neighbor sharing the matching content category c_2 is chosen for forwarding the query. Among the rest of the neighbors, peer c_6 is chosen randomly (since two forwarding peers are to be selected).

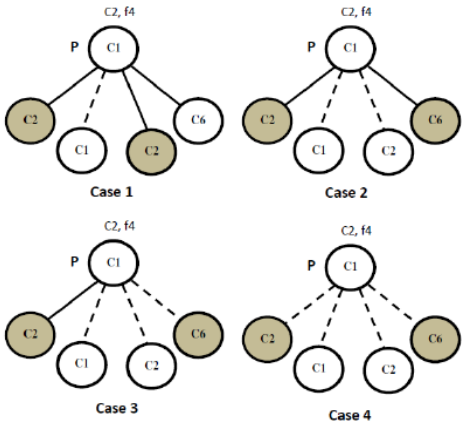

Figure 1. Neighbor selection by peer P for forwarding the query string (c_2, f_4). The community edges and the connectivity edges are drawn using solid and dotted lines respectively. The peers that receive the query for forwarding are shaded.

3.2.1.2 Query forwarding

The query forwarding procedures has four steps. (i) *Check the trust level of peer j:* Peer i checks trust rating of peer j through check trust rating algorithm (explained in Section 3.2.3). Selection of peers for further forwarding the query is made accordingly. (ii) *Check the availability of file*: If the requested file is found, response is sent to peer j. If TTL value has not expired, the following steps are executed. (iii) *Calculate the number of messages to be sent*: It is calculated based on the value of $Prob_{com}$. (iv) *Choose neighbors*: Neighbors are chosen in using neighbor selection rule. The search process is shown in Fig. 2. It is assumed that the query is forwarded at each hop to two neighbors. The matching community links are preferred over connectivity links to dispatch query. Peer 1 initiates the query and forwards it to two community neighbors 3 and 4. The query reaches peer 8 via peer 4. However, peer 8 knows from its previous transactions with peer 4 that peer 4 is malicious. Hence it blocks the query. The query forwarded by peer 5 is also blocked by peer 10 and 11 as both of them know that peer 5 is malicious. The query is matched at four peers: 4, 6, 9 and 12. The search process is shown in Fig. 2.

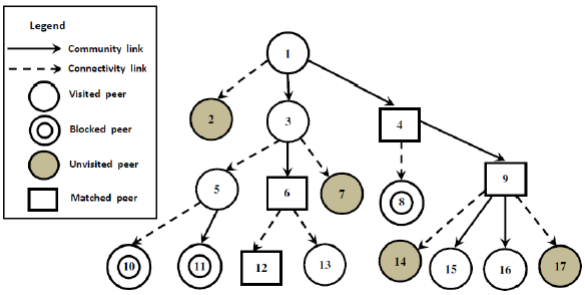

Figure 2. The breadth first search (BFS) tree for the search initiated by peer 1.

When a query reaches to peer i from peer j, peer i forwards the query further in the network as discussed below.

3.2.2 Topology adaptation

The responses are sorted by the initiating peer i based on the reputations of the resource providers and the peer having the highest reputation is selected as the source for downloading. The requesting peer checks the authenticity of the downloaded file. If the file is found to be fake, peer i attempts to download the file from other sources until it is able to find the authentic resource or no more sources exist for searching. The peer then updates the trust ratings and possibly adapts the network topology after failed or successful download, to bring trusted peers to its neighborhood and to drop malicious peers from its community. The restructuring of network is controlled by a parameter known as *degree of rewiring* which provides the probability with which a link is formed between a pair of peers. This parameter allows trust information to propagate through the network. The topology adaptation consists of the following operations: (i) *link deletion*: Peer i deletes the existing community link with peer j if it finds peer j as malicious. (ii) *link addition*: Peer i probabilistically forms a community link with peer j if the resource provided by the peer j is found to be authentic. If $RIC \leq edge\_limit$, for both peers i and j, only then an edge can be added subject to the approval of resource provider j. If peer j finds that peer i is malicious (i.e., its trust value is below the threshold), it doesn't approve the link.

Fig. 3 illustrates topology adaptation on the network topology shown in Fig. 2. In the example shown in Fig. 3, peer 1 downloads the file from peer 4 and finds that the file is spurious. It reduces the trust score of peer 4 and deletes the community link 1-4. It then downloads the file from peer 6 and gets an authentic file. Peer 1 now sends a request to peer 6, and the latter grants the request after checking its trust value and the community edge 1-6 is added. The malicious peer 4 loses one community link and peer 6 gains one community edge. However, the network still remains connected by connectivity edges, shown in dotted lines.

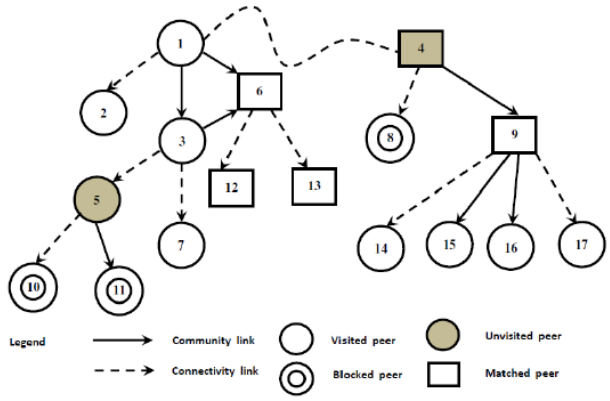

Figure 3. Topology adaptation based on outcome of the search in Figure 2. The malicious nodes are shaded in gray color.

3.2.3 Checking of trust rating of peers

Trust rating is used at various stages of execution of the algorithm to take various decisions such as: possible source for download, to stop a query forwarded from a malicious node and to adapt the topology. A *least recently used* (LRU) data structure is used at each peer to keep track of 32 most recent peers it has interacted with. When no transaction history is available, a peer seeks recommendation from its neighbors using *trust query*. When peer *i* doesn't have trust score of peer *j* in its LRU history, it first seeks recommendation about *j* from all of its community neighbors. If none of its community neighbors possesses any information about *j*, peer *i* initiates a *directed DFS search*. The trust computation model has been presented in Section 3.1.4.

3.2.4 User privacy protection in searching

User privacy has emerged as a very critical issue in many of the network-based applications [21, 22, 23]. The trust-based searching scheme described above does not guarantee any privacy requirement of the requester (i.e. the initiator of the query). For protecting the privacy of the user, several enhancement of the algorithm are proposed. Following cases are identified for privacy preservation.

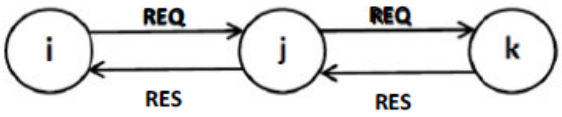

Figure 4. Identity protection of the requesting peer *i* from the supplier peer *k* by use of trusted peer *j*. REQ and RES are the request and response message respectively.

3.2.4.1 Protection of the identity of the requesting peer

In this case, as shown in Fig. 4, instead of sending the request straightway to the supplier peer, the requesting peer asks one of its trusted peers (which may or may not be its neighbor) to look up the data on its behalf. Once the query propagation module successfully identifies the possible supplier of the resource, the trusted peer serves as a proxy to deliver the data to the requester node. Other peers including the supplier of the resource will not be able to know the real requester. Hence, the requester's privacy is protected. Since the requestor's identity is only known to its trusted peer, the strength of privacy is dependent on the effort required to compromise the trusted peer. As mentioned in Section 3.1.5, the message communicated the peers are encrypted by 1024 bit RSA key, which is a provably secure algorithm. Hence, the privacy of the requester is highly protected.

3.2.4.2 Protecting the data handle

To improve the achieved privacy level, the data handle may not be put in the request at the beginning. When a requester initiates the request, it computes the hash value of the handle and reveals only a part of the hash result in the request sent to its trusted peer. The steps 1 and 2 in Fig. 5 represent these activities. Each peer receiving the request compares the revealed partial hash to hash codes of the data handles that it holds. Depending on the length of the revealed part, the receiving peer may find multiple matches. This does not, however, imply that the peer has the requested data. Thus this peer will provide a candidate set, along with a certificate of its public key, to the requester. If the matched set is not empty, the peer will construct a *Bloom filter* [2] based on the left parts of the matched hash codes, and send it back to the trusted peer. The trusted peer forwards it back to the requester. These are represented by the steps 3 and 4 in Fig. 5. Examining the filters, the requester can eliminate from the candidate data supplier list all peers that do not have the required data. It then encrypts the complete request with the supplier's public key and gets the requested data with the help from its trusted node. The steps 5, 6, 7 and 8 in Fig. 5 represent these activities. By adjusting the length of the revealed hash code, the requestor can control the number of eliminated peers. The level of privacy is improved manifold since the malicious peers need to both compromise the trusted node and break the Bloom filter and hash function.

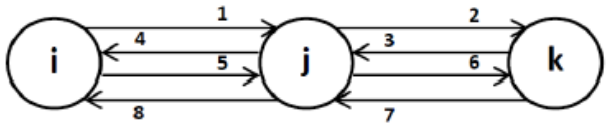

Figure 5. Protecting data handle using trusted node. Peer *i* and *k* are the requester and the supplier peer respectively. Peer *j* is the trusted peer of the requester peer *i*.

3.2.4.3 Hiding the data content

Although the privacy-preservation level has been improved during the look-up phase using the previous two schemes, the privacy of the requester will be compromised if the trusted node can see the data content when it relays the packets for the requester. To improve the privacy level and prevent eavesdropping, we can encrypt the data handle and the data content. If the identity of the supplier is known to the requester, it can encrypt the request using the supplier's public key. The public key of the requester cannot be used because the certificate will reveal its identity. The problem is solved in the following manner. The requester generates a symmetric key and encrypts it using a supplier's public key. Only the supplier can recover the key and use it to encrypt

data. To prevent the trusted node of the requester from conducting a man-in-the-middle attack, the trusted node is required to sign the packet. This provides non-repudiation evidence, and shows that the packet is not generated by the trusted node itself. The privacy level has been improved since now the malicious nodes need to break the encryption keys as well.

4. Performance Evaluation

To analyze the performance of the proposed algorithm, several metrics are first defined. An extensive evaluation of the performance of the proposed scheme is then made based on these metrics.

4.1 Attempt ratio (AR)

A peer keeps on downloading files from various sources based on their trust rating till it gets the authentic file. AR is the probability that the authentic file is downloaded in the first attempt. A high value of AR is desirable for a searching scheme to be efficient and scalable.

4.2 Effective attempt ration (EAR)

It measures the cost of downloading an authentic file by a good peer in comparison to the cost incurred by a malicious peer. If $P(i)$ be the total number of attempts made by the peer i to download an authentic file, EAR is given by (8).

$$EAR = \left(\frac{1}{M}\sum_{i=1}^{M}\frac{1}{P(i)} - \frac{1}{N}\sum_{j=1}^{N}\frac{1}{P(j)}\right) \tag{8}$$

In (8), M and N are the number of malicious and good peers issuing queries in a particular generation. For example, EAR = 50 implies that if a good peer needs one attempt to download an authentic file, a malicious peer will need two attempts.

4.3 Query miss ratio (QMR)

Since the formation of semantic communities takes some time, there will be a high rate of query misses in the first few generations of search. However, as the algorithm executes, the rate of query miss is expected to fall for the good peers. QMR is defined as the ratio of the number of search failures to the total number of searches in a generation.

4.4 Hit per message (HM)

Due to the formation of semantic communities in the network, number of messages required to get a hit is expected to fall down as the network topology stabilizes. HM measures the search efficiency achieved by the proposed search algorithm and it is defined as the number of query hits per message irrespective of the authenticity of the file being downloaded.

4.5 Relative increase in connectivity (RIC)

After a successful download, a requesting peer attempts to establish a community edge with the resource provider, if approved by the latter. This ensures that peers which provide good community services are rewarded by having increasing number of community neighbors. The metric RIC measures the number of community neighbors a peer gains with respect to its connectivity neighbors in the initial network topology. If $D_{init}(i)$ and $D_{final}(i)$ are the initial and final degrees of the peer i, and N is the number of peers, then RIC for peer i is computed using (9).

$$RIC(i) = \frac{1}{N}\sum_{i}\frac{D_{final}(i)}{D_{init}(i)} \tag{9}$$

4.6 Closeness centrality (CCen)

Since the topology adaptation effectively brings the good peers closer to each other, the length of the shortest path between a pair of good peers decreases. This intrinsic incentive for sharing authentic files is measured by the metric CCen. The peers with higher CCen values are topologically better positioned. If P_{ij} is the length of the shortest path between peer i and peer j through the community edges and if V denotes the set of peers, then CCen for peer i is given by (10).

$$CCen(i) = \frac{1}{\sum_{j \varepsilon V} P_{ij}} \tag{10}$$

4.7 Clustering coefficient (CC)

It gives an indication about how well the network forms cliques. It has an important role in choosing the TTL value in the search algorithm. With higher values of CC, lower TTL values can be used. If K_i be the number of community neighbors of peer i, then clustering coefficient (CC) of peer i is computed using (11).

$$CC(i) = \frac{2E_i}{K_i(K_i - 1)} \tag{11}$$

In (11), E_i is the actual number of community edges between the K_i neighbors. CC of the network is taken as the average value of all *CC(i)*s.

4.8 Largest connected component (LCC)

The community edges connect nodes which have similar content interests and sufficiently high mutual trust between each other. If we consider the peers which share a particular category of contents, then the community edges form a trust-aware community overlay. However, it will be highly probable that the trust-aware overly graph will be a disconnected graph. LCC is the largest connected component of this disconnected overlay graph. LCC of the network can be taken as a measure of the goodness of the community structure since it signifies how strongly the peers with similar contents and interests are connected with each other.

4.9 Trust query propagation overhead (TQPO)

The peers build trust and reputation information both by collection of first-hand and second-hand information. Trust query message is propagated when trust information about a peer is not available locally in a peer. A trust query message involves one DFS round without backtracking. The overhead incurred due to trust query propagation is measured by the metric called *trust query propagation overhead* (TQPO).

TQPO is defined as the total number of distinct DFS search attempts per generation. It may be noted that a trust query may be initiated multiple times for a single file search operation: to select a trusted neighbor or to approve a community link.

4.10 Topology adaptation overhead (TAO)

It gives an idea about the overhead due to the topology adaptation and is measured by the number of community edges added or deleted in a generation of network simulation. Larger number of addition and deletion of community edges will lead to increased overhead.

4.11 Simulation results

A discrete time simulator written in C is used for simulation. In simulation, 6000 peer nodes, 18000 *connectivity edges*, 32 *content categories* are chosen. The *degree of deception* and the *degree of rewiring* are taken as 0.1 and 0.3 respectively. The value of the *edge_limit* is taken as 0.3. The TTL values for BFS and DFS are taken as 5 s and 10 s respectively. The discrete time simulator simulates the algorithm repeatedly on the power law network and outputs all the metrics averaged over generations. *Barabasi-Albert* generator is used to generate initial power law graph with 6000 nodes and approximately 18000 edges. The number of search per generation is taken as 5000 while the number of generations per cycle of simulation is 100.

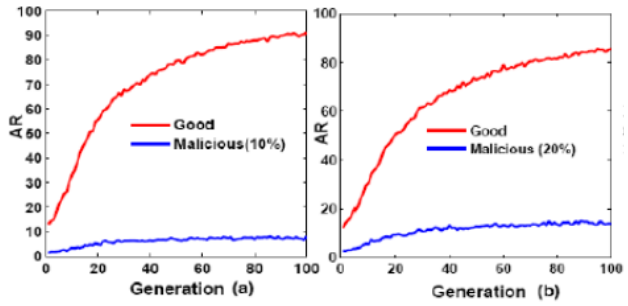

Figure 6. AR vs. percentage of malicious peers. In (a) 10%, and in (b) 20% nodes in the network are malicious.

To check the robustness of the algorithm against attacks from malicious peers, the percentage of malicious peers is gradually increased. Fig. 6 illustrates the cost incurred by each type of peers to download authentic files. As the percentage of malicious peers is increased, cost incurred by malicious peers to download authentic files decreases while that of good peers increases. This is illustrated in Fig. 7 using EAR.

It is evident from Fig. 7 that when 10% of peers in the network are malicious peers, EAR is 80; i.e., on the average, if a good peer needs one attempt to download an authentic file, a malicious peer needs 5 attempts to do so. The peers who share high quality files acquire good reputation and earn more community edges and eventually disseminates query through the community edges only. The queries are forwarded via trusted peers at each hop, and hence the probability of getting authentic files in the first attempt increases. However, as the queries forwarded by malicious peers are blocked by good peers, they need more attempts to download good files. As the percentage of malicious peers in the network increases, EAR drops to 20. Up to 60% malicious peers in the network, the good peers have higher probability to get authentic files in their first attempts. It is clear that the proposed algorithm can withstand attacks by malicious peers as long as the percentage of malicious peers in the network does not exceed 60.

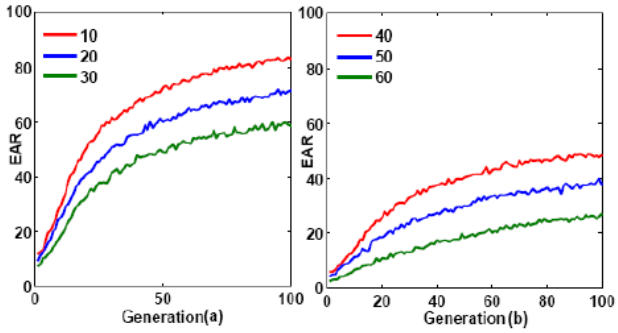

Figure 7. EAR vs. percentage of malicious peers. In (a) 10% and in (b) 20% nodes are malicious.

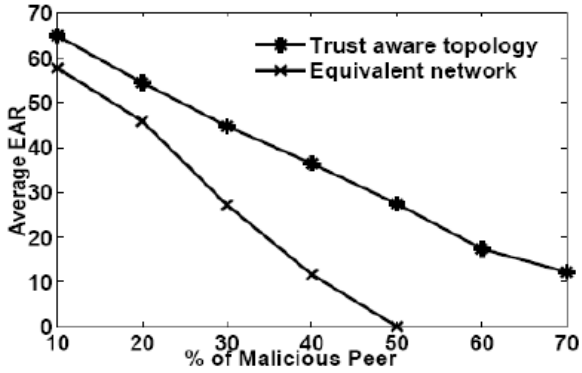

Figure 8. Avg. EAR vs. percentage of malicious peers in networks with and without the trust management scheme.

The performance of the proposed protocol is compared with an equivalent power law network with no trust management. Since the proposed algorithm allows addition of community edges, therefore, to keep the number of edges in both networks equal, additional edges are introduced between similar peers in the equivalent network. Fig. 8 shows the comparison of the average EAR values. In the network without trust and reputation management, EAR drops to zero when at least 50% of nodes in the network are malicious. However, in the network with trust management (i.e. with the proposed protocol), even with 60% malicious peers, EAR is consistently sustained at 20. This clearly demonstrates the robustness of the proposed protocol.

Fig. 9 shows QMR experienced by both types of peers for varying percentages of malicious peers in the network. Initially, QMR is high as no interest-based communities are formed and the searching is essentially a blind (i.e., brute force) one. As the algorithm executes further, the peers with similar content interests come closer to each other (in terms of number of hops between them), and the queries are forwarded through the community edges. As a result, QMR drops for good peers. It is observed from Fig. 9 that the steady state value of QMR for good peers is less than 0.2,

and QMR is independent of the percentage of malicious peers in the network. This is a significant performance achievement of the proposed algorithm. For malicious peers, the steady state value of QMR is 0.4. The high QMR for malicious peers is due to the fact that the queries from the malicious peers are blocked by the good peers. It is evidently clear from the results that the proposed algorithm effectively rewards the peers which share large number of authentic files in the network which in turn helps in making the searching process efficient.

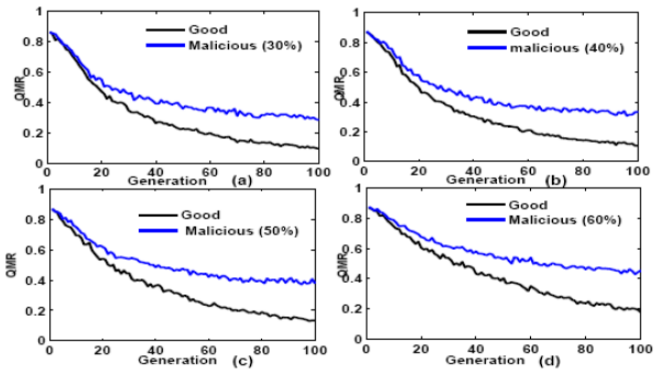

Figure 9. QMR for various percentages of malicious peers in the network.

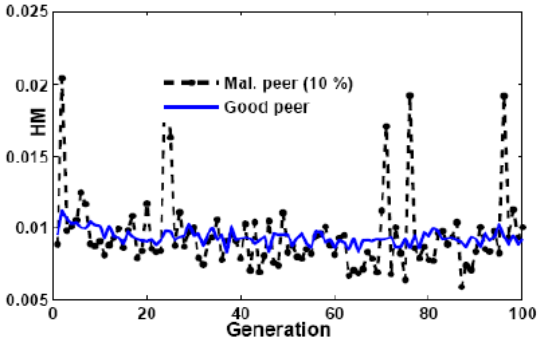

Figure 10. HM for malicious and honest peers in the network. Percentage of malicious peers in the network is 10.

Fig. 10 shows variation of HM for both types of peers. Although, the value of HM for good peers reaches a steady state as the topology matures, for malicious peers, the value of HM fluctuates quite appreciably. HM for malicious peers sometimes attains higher values than the good peers. Since the queries forwarded by the malicious peers are blocked, HM for these peers are sometimes higher than the good peers. The *hit* here does not mean authentic hit. The authentic hit of good peers is higher than that of malicious peers as these peers have higher AR values.

Fig. 11 shows the variation of RIC for each type of peers under threat model *A*. It may be observed that RIC for good peers increases to 2.4 (constrained by the edge limit), whereas for malicious peers, RIC does not increase beyond 1.2. With the increase in the percentage of malicious peers, the saturation rate slows down albeit the final value remains the same. This shows that the proposed algorithm provides better connectivity to the peers which share large number of authentic files. At the same time the malicious peers are blocked gradually and their connectivities in terms of community edges are reduced.

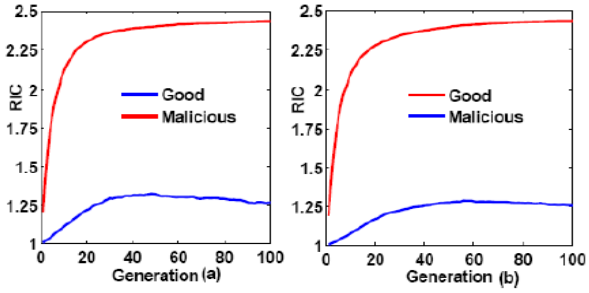

Figure 11. RIC for various percentages of malicious peers under *threat model A*. In (a) 20% and in (b) 40% peers in the network are malicious.

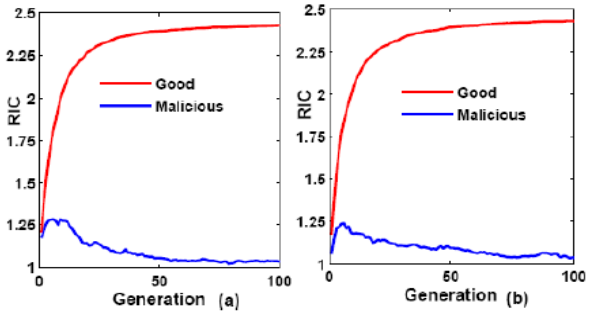

Figure 12. RIC for various percentages of malicious peers under *threat model B*. In (a) 20% and in (b) 40% peers in the network are malicious.

Fig. 12 shows the variation of RIC under threat model *B*. Since in this model, a malicious peer starts providing fake files (i.e., starts acting maliciously) after achieving high connectivity because of its good behavior in the past, and again stops acting maliciously after it loses a larger number of connecting edges, fluctuation in RIC persists throughout the simulation period.

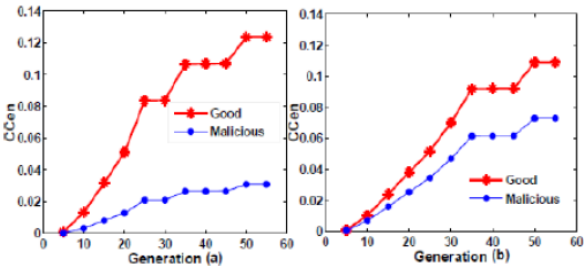

Figure 13. Closeness centrality for various percentages of malicious nodes. In (a) 20% and in (b) 40% nodes are malicious.

Fig. 13 presents how the *closeness centrality* (CCen) of good and malicious peers varies in the community topology. In computation of CCen, only the community edges have been considered. It may be observed that the steady state value of CCen for good peers is around 0.12. However, for the malicious peers, the CCen value is found to lie in between 0.03 to 0.07. This demonstrates that the malicious peers are driven to the fringe of the network, while the good peers are allowed to form communities.

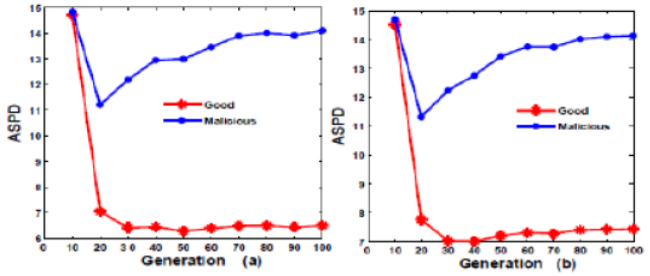

Figure 14. Avg. shortest path distance vs. generations of search at the step of ten for various percentages of malicious peers. In (a) 30% and in (b) 40% nodes are malicious.

Higher values of CCen also indicate that good peers have smaller average shortest path length between them. In the simulation, the diameter of the initial network is taken as 5. At the completion of a simulation run, if there is no path between a pair of peers using community edges, then the length of the shortest path between that pair is assumed to be arbitrarily long, say 15 (used in Fig. 14). As shown in Fig. 14, the *average shortest path distance* (ASPD) decreases form the initial value of 15 for both honest and malicious nodes. However, the rate and the extent of decrease for the good peers are much higher due to the formation of semantic communities around them. For malicious peers, after an initial fall, the value of ASPD increases consistently and finally almost reaches the maximum value of 15. On the other hand, the average value of ASPD for good peers is observed to be around 6. Since the good peers are connected with shorter paths, the query propagations and their responses will also be faster among these peers.

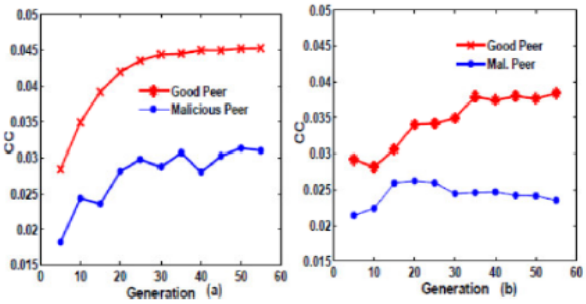

Figure 15. Clustering coefficient for different percentages of malicious peers. In (a) 20% and in (b) 40% of the peers are malicious.

Fig. 15 shows *clustering coefficient* (CC) for each type of peers. Since community edges are added based on the download history and peers having good reputation gain more community edges, clustering coefficient (CC) is high for good peers. This leads to triangle formation in the communities. To counter this phenomenon, the search strategy adapts itself from BFS to DFS to minimize redundant message flows in the network. Since edges are added based on the download history and similarity of interest, community of peers are formed which are connected to other community by hub of peers having interest in multiple content categories. This leads to lower ASPD for good peers.

Fig. 16 depicts the size of the *largest connected component* (LCC) for each of the 32 content categories. It may be observed that the average size of LCC for all content categories remains constant even if the percentage of malicious peers in the network increases. This clearly shows that the community formation among the good peers is not adversely affected by the presence of malicious peers.

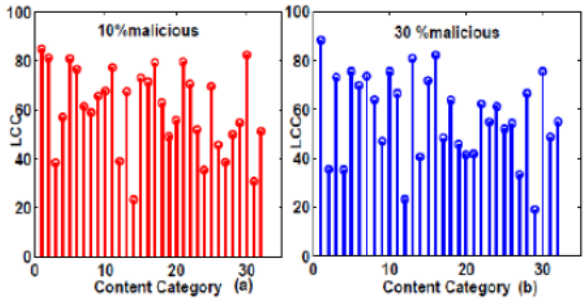

Figure 16. Largest connected components (LCCs) for different content categories.

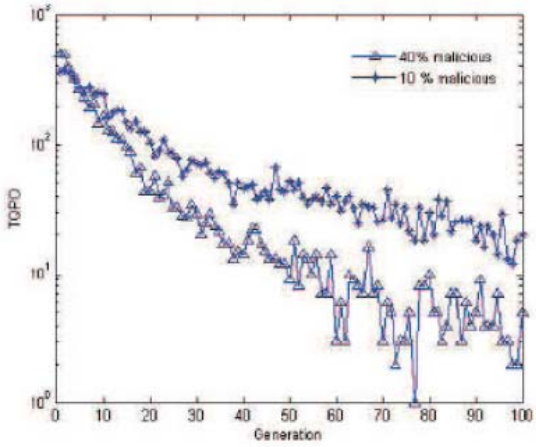

Figure 17. Overhead of trust query propagation for 10% and 20% malicious peers in the network.

Fig. 17 shows that as the topology of the network matures, the steady state value of *trust query propagation overhead* (TQPO) attains a quite low value. The value of TQPO is less than 10 when 10% of the peers in the network are malicious. Even when the network has 40% of its peers malicious, TQPO gradually decreases and reaches a value of 20 within the span of 100 generations. Hence, the trust propagation module has little impact on the system overhead since the trust information is efficiently distributed in the trust-aware overlay topology.

Finally, the overhead due to the topology adaptation in the proposed scheme is investigated. As mentioned in Section 4.10, the overhead due to the topology adaptation is measured by the metric TAO, which is defined as the number of community edges added or deleted in a generation. Fig. 18 shows the variation of TAO for different percentages of malicious peers. It is observed that TAO starts falling from an initial high value and oscillates with small amplitudes. This is due to the fact that initially the edge capacities of the peers are not saturated and they acquire community edges rapidly. As the algorithm executes, the good peers acquire relatively stable neighborhood resulting in a sharp decrease in the value of TAO. In the

subsequent generations, the value of TAO fluctuates slightly since the good nodes delete the existing edges with malicious peers as soon as the malicious peers are detected, and acquire new community edges with the fellow good peers. With the increase in percentage of malicious peers, fluctuations in the values of TAO also increase as increasing number of nodes get added and deleted in the network. However, in all cases, the value of TAO falls sharply and attains a very low value once the community topology of the peers become stable. This shows that the proposed algorithm introduces a very small overhead in computation for topology adaptation.

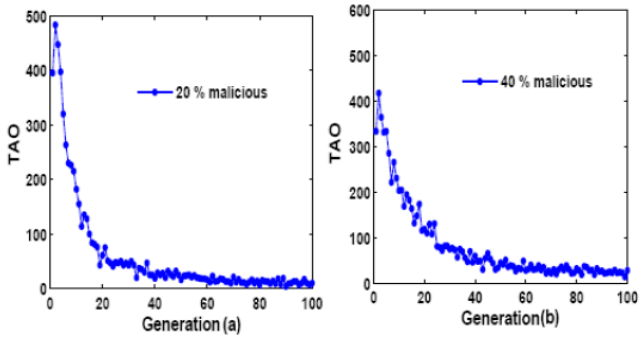

Figure 18. Overhead due to topology adaptation under the presence of various percentages of malicious peers. In (a) 20% and in (b) 40% of the peers in the network are malicious.

4.12 Comparison with existing schemes

In the following, we provide a brief comparative analysis of the proposed protocol with two similar protocols existing in the literature. In [9], a method to minimize the impact of malicious peers on the performance of a peer-to-peer system has been proposed, where the global trust value for each peer is computed by calculating the left principal Eigen-vector of a matrix of normalized local trust value. Since the trust and reputation computations are robust, the mechanism is able to sustain a high value of AR (i.e. the fraction of authentic download) for good peers even when the percentage of malicious peers is as high as 80. In contrast, the proposed protocol in this paper can support high value of AR for good nodes as long as the percentage of malicious peers in the network does not exceed 60. However, the scheme based on Eigen trust is computationally very intensive and it is susceptible to produce unreliable results in case of Byzantine failure of some peers. On the other hand, the proposed trust management algorithm in this paper is light-weight, and it can efficiently identify free riding and Byzantine failure of peers while improving on the QoS of searching. In the APT protocol [3], as the topology stabilizes, all the paths from the good peers to the malicious peers are blocked, and the characteristic path lengths of these two types (good and malicious) of peers are distinctly different. However, in the proposed protocol in this paper, paths still exist between good peers and the malicious peers through the connectivity edges in the original network. These connectivity edges are deleted during the execution of the algorithm. This prevents any possibility of network partitioning thereby making the protocol more robust. Moreover, the scalability of the proposed protocol is higher than that of the APT protocol, since it uses a light-weight trust management module. More importantly, the APT protocol does not have any mechanism to protect user and data privacy. This makes the protocol impractical in real-world deployment scenario where user privacy is a critical issue. The proposed protocol provides a very robust and reliable mechanism for protection of both user and data privacy, which makes it more attractive for deployment in real-world peer-to-peer networks.

5. Conclusion

In this paper, a search mechanism is proposed that solves multiple problems in peer-to-peer networks e.g., inauthentic download, poor search scalability, combating free riders and protecting user privacy. It is shown that by topology adaptation, and robust trust management, it is possible to isolate the malicious peers while providing topologically advantageous positions to the good peers so that good peers get faster and authentic responses to their queries. A large number of metrics are defined for evaluating the performance of the proposed scheme. The protocol is simulated on a power-law network. The simulation results have demonstrated that the protocol is robust even in presence of a large percentage of malicious peers in the network. A brief comparative analysis of the scheme has been made with some of the well-known similar schemes existing in the literature. As a future plan of work, we intend to carry out an analysis of the message overhead of the privacy module and a detailed comparative study of the performance results of the proposed protocol in this paper with other existing searching mechanisms for peer-to-peer networks.

References

[1] A. Abdul-Rahman, S. Hailes, “A distributed trust model,” in *Proceedings of the Workshop on New Security Paradigms*, pp. 48-60, 1997.

[2] B. Bloom, “Space-time trade-offs in hash coding with allowable errors,” *Communications of the ACM*, 13(7), 422 – 426, 1970.

[3] T. Condie, S. D. Kamvar, and H. Garcia-Molina, “Adaptive peer-to-peer topologies,” in *Proceedings of the 4th International Conference on Peer-to-Peer Computing (P2P’04)*, pp. 53-62, 2004.

[4] A. Crespo and H. Garcia-Molina, “Semantic overlay networks for p2p systems,” Technical Report, Stanford University, 2002.

[5] E. R. De Mello, A. V. Moorsel, and J. D. S. Fraga, “Evaluation of p2p search algorithms for discovering trust paths,” in *Proceedings of the 4th European Performance Engineering Conference on Formal Models and Stochastic Models for Performance Evaluation*, pp. 112- 124, 2007.

[6] S. Ganeriwal and M. Srivastava, “Reputation-based framework for high integrity sensor networks,” in *Proceedings of the 2nd ACM Workshop on Security of Ad Hoc and Sensor Networks (SAN ’04)*, pp. 66 – 77, 2004.

[7] L. Guo, S. Yang, K. Shen, and W. Lu, “Trust-aware adaptive p2p overlay topology based on super-peer-

partition," in *Proceedings of the 6th Int. Conf. on Grid and Cooperative Computing*, pp. 117-124, 2007.

[8] A. Jsang, "A logic for uncertain probabilities," *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol 9, no 3, pp. 279 – 311, 2001.

[9] S. D. Kamvar, M. T. Schlosser, and H. Garcia-Molina, "The Eigen trust algorithm for reputation management in p2p networks," in *Proceedings of the 12th International Conference on World Wide Web (WWW'03)*, 2002.

[10] X. Li and J. Wang, "A global trust model of p2p network based on distance-weighted recommendation," in *Proceedings of IEEE International Conference of Networking, Architecture, and Storage*, pp. 281-284, 2009.

[11] J. Risson, T. Moors, "Survey of research towards robust peer-to-peer networks," *Computer Networks*, vol 50, no 7, pp. 3485-3521, 2006.

[12] J. Schafer, K. Malinks, P. Hanacek, "Peer-to-peer networks security," in *Proceedings of the 3rd International Conference on Internet Monitoring and Protection (ICIMP)*, pp. 74-79, 2008.

[13] M. T. Schlosser, T. E. Condie, S. D. Kamvar, A. D. Kamvar, "Simulating a p2p file-sharing network," in *Proceedings of the 1st Workshop on Semantics in P2P and Grid Computing*, 2002.

[14] J. Sen, "A trust-based robust and efficient searching scheme for peer-to-peer networks", in *Information and Communications Security*, Soriano et al. (eds.), pp. 77-91, LNCS vol 6476, Springer-Verlag, Heidelberg, Germany, 2010.

[15] J. Sen, "A secure and efficient searching scheme for trusted nodes in a peer-to-peer network," in *Computational Intelligence in Security for Information Systems*, Herrero, A. et al. (eds.), pp. 100 - 108, LNCS, vol 6694, Springer-Verlag, Heidelberg, Germany, 2011.

[16] J. Sen, P. R. Chowdhury, and I. Sengupta, "A distributed trust mechanism for mobile ad hoc networks," in *Proceedings of the International Symposium on Ad Hoc and Ubiquitous Computing (ISAHUC'06)*, Surathkal, Mangalore, India, pp. 62-67, 2006.

[17] J. Sen, P. R. Chowdhury, and I. Sengupta, "A distributed trust establishment scheme for mobile ad hoc networks," in *Proceedings of the International Conference on Computation: Theory and Applications (ICCTA'07)*, pp. 51-57, Kolkata, India.

[18] J. Sen, "A distributed trust and reputation framework for mobile ad hoc networks," in *Proceedings of the 1st International Conference on Networks Security and its Applications (CNSA'10)*, Chennai, India, pp. 538-547, Springer Communications in Computer and Information Science (CCIS), vol 89, Springer-Verlag, Heidelberg, Germany, 2010.

[19] J. Sen, "A trust-based detection algorithm of selfish packet dropping nodes in a peer-to-peer wireless mesh network," in *Proceedings of the 1st International Conference on Networks Security and its Applications (CNSA'10)*, Chennai, India, pp. 528-537, Springer Communications in Computer and Information Science (CCIS), vol 89, Springer-Verlag, Heidelberg, Germany, 2010.

[20] J. Sen, "Reputation- and trust-based systems for wireless self-organizing networks," book chapter in *Security of Self-Organizing Networks: MANET, WSN, WMN, VANET*, Al-Shakib Khan Pathan (ed.), Aurbach Publications, CRC Press, Taylor & Francis group, USA, 2010.

[21] J. Sen, "Privacy preservation technologies in Internet of Things," in *Proceedings of the International Conference on Emerging Trends in Mathematics, Technology and Management*, pp. 496-504, Shantiniketan, India, 2010.

[22] J. Sen, "An efficient and user privacy-preserving routing protocol for wireless mesh networks," *International Journal of Scalable Computing: Practice and Experience, Special Issue on Network and Distributed Systems*, vol 11, no 4, pp. 345-358, 2010.

[23] J. Sen, "An anonymous authentication and communication protocol for wireless mesh networks," in *Proceedings of the 1st International Conference on Advances in Computing and Communications (ACC'11),* Abraham, A. et al. (eds), Springer CCIS vol 193. part VI, pp. 581-592, Kochi, India, 2011.

[24] G. Shafer, *A Mathematical Theory of Evidence,* Princeton University, 1976.

[25] H. Tain, S. Zou, W. Wang, and S. Cheng, "Constructing efficient peer-to-peer overlay topologies by adaptive connection establishment," *Computer Communication*, vol 29, no 17, pp. 3667-3579, 2006.

[26] Y. Tang, H. Wang, W. Dou, "Trust based incentive in p2p network," in *Proceedings of the IEEE International Conference on E-Commerce Technology for Dynamic E-Business*, pp.302-305, 2004.

[27] L. Xiao, Y. Liu, N. Lionel, "Improving unstructured peer-to-peer systems by adaptive connection establishment," *IEEE Transaction on Computers*, vol 54, no 9, pp. 1091-1103, 2005.

[28] H. Zhuge, X. Chen, and X. Sun, "Preferential walk: towards efficient and scalable search in unstructured peer-to-peer networks," in *Proceedings of the 14th Int. Conf. on World Wide Web (WWW'05),* pp. 882-883, 2005.