SocialLink: Utilizing Social Network and Transaction Links for Effective Trust Management in P2P File Sharing Systems

Kang Chen
Department of Electrical and Computer Engineering
Southern Illinois University, Carbondale, IL 62901
Email: kchen@siu.edu

Guoxin Liu, Haiying Shen, and Fang Qi
Department of Electrical and Computer Engineering
Clemson University, Clemson, SC, USA 29631
Email: {guoxinl, shenh, fqi}@clemson.edu

***Abstract*—Current reputation systems for peer-to-peer (P2P) file sharing systems either fail to utilize existing trust within social networks or suffer from certain attacks (e.g., free-riding and collusion). To handle these problems, we introduce a trust management system, called SocialLink, that utilizes social network and historical transaction links. SocialLink manages file transactions through both the social network and a novel weighted transaction network, which is built based on previous file transaction history. First, SocialLink exploits the trust among friends in social networks by enabling two friends to share files directly. Second, the weighted transaction network is utilized to 1) deduce the trust of the client on a server in reliably providing the requested file and 2) check the fairness of the transaction. In this way, SocialLink prevents potential misbehaving transactions (i.e., providing faulty files), encourages nodes to contribute file resources to non-friends, and avoids free-riding. Furthermore, the weighted transaction network helps SocialLink resist whitewashing, collusion and Sybil attacks. Extensive simulation demonstrates that SocialLink can efficiently ensure trustable and fair P2P file sharing and resist the aforementioned attacks.**

I. Introduction

Peer-to-peer (P2P) file sharing systems are prone to have selfish or malicious nodes due to their open and distributed environment. For example, without a fairness mechanism, free-riders [1] that hardly provide files to others can still receive files from other nodes in the system. Without a central access control, malicious users can distribute corrupted files or files containing viruses (i.e., faulty files) easily. Previous researches have shown that 85% of Gnutella users are selfish users sharing no files, and 44% of files downloaded through the Kazaa file sharing application in a test contained malicious code [2], [3]. Such selfish and malicious behaviors can greatly degrade the stability and efficiency of P2P file sharing systems. Consequently, cooperation incentives are needed to encourage cooperative behaviors and discourage misbehaviors in P2P file sharing systems.

Reputation (or trust) management systems, as a cooperation incentive method for P2P file sharing systems, have been widely studied in recent years [4]–[9]. As implemented in online market platforms (i.e., eBay [10], Amazon [11] and Overstock [12]), reputation systems compute the global reputation/trust value for each user based on collected ratings. When a client node requests for a service, it queries the reputation system for the reputation values of candidate servers and chooses the one with the highest reputation as the server. Nodes with reputation values lower than a threshold are considered as untrustworthy and their service requests are rejected. However, these systems suffer from attacks such as free-riding, whitewashing, collusion and Sybil attack. A free-rider can maintain its reputation slightly higher than the threshold to always receive files without providing files to others. In whitewashing, a low-reputed node can simply abandon its account and create a new account with the initial reputation to receive services. Colluders/Sybil nodes purposely provide good feedbacks to one or more nodes to increase their reputations. Such boosted reputations can be exploited for malicious behaviors, e.g., disseminating faulty files in the system.

Recently, a number of systems [13]–[21] have been proposed for efficient and trustable P2P services by exploiting the social network properties [22]–[24]. In these systems, nodes seek for services from friends in the social network directly since social friends often are trustable. However, the social network of a user usually only contains a small number of users in system. As a result, a client may not be able to find a requested file from its friends, i.e., limited file availability. Therefore, such a social network based reputation system must be complemented by a reputation system to achieve global file availability for reliable file sharing. Meanwhile, a mechanism is also needed to prevent common attacks in normal reputation systems.

To leverage the social networks and meanwhile overcome the shortcomings of existing reputation systems, we propose SocialLink, a social network based trust management system that enables nodes to receive re-

liable file services from both friends and non-friends. It can effectively resist free-riding, whitewashing, collusion and Sybil attacks.

File sharing between friends. Specifically, SocialLink allows each user to maintain a social network consisting of reliable users including both real-world friends and frequently interacted nodes in the file sharing system (i.e., online-friends). Given a number of server options, a user chooses his/her social friends directly, if available, since friends are often trustable. By doing this, file sharing can be conducted efficiently without reputation querying. The social network based file sharing encourages users to be continuously cooperative and discourages selfish behaviors because 1) people do not want to damage their real-life reputations among friends and 2) users would like to have more online friends for more file resources from friends.

File sharing between non-friends. For reliable file sharing between non-friends, SocialLink maintains a *weighted transaction network*. When server N_i provides a file to client N_j for the first time, a link is created from N_i to N_j with a link weight that equals to the file size. The weight is updated after each transaction that has utilized the trust represented by the link. Suppose the weights of link $N_i \rightarrow N_j$ and $N_j \rightarrow N_i$ are w_1 and w_2, respectively. This means that N_i has the trust to provide file with size w_1 to N_j and N_j has the trust to provide file with size w_2 to N_i, respectively. Such information is utilized to ensure the reliability and fairness of the file transaction between N_i and N_j.

Since non-friends may be connected by a path consisting of a number of links, we define the weight of a path as the minimal link weight, which shows the path's trust on file provision. For a client and an identified server, their *trust-flow* is defined as the maximal one among the weights of all server-client paths, and their *upload-flow* is defined as the maximal one among the weights of all client-server paths. Then, SocialLink requires that the trust-flow must be larger than the requested file size to prevent faulty file dissemination or the difference between trust-flow and upload-flow must be within a reasonable range to prevent free-riding. A transaction that does not meet the two requirements is not allowed.

In addition to the free-riding, SocialLink can also prevent whitewashing, collusion and Sybil attacks to a certain degree. When a whitewasher creates a new account, it must provide files in order to receive files from non-friends. Hence, the whitewash does not help free-riding. Though a collective of colluders can boost their own reputations, they still need to conduct actual transactions with outside nodes to build connections with them to receive files from them.

In summary, the major contributions of SocialLink are summarized as the following:

- Exploiting the social network property to enable efficient and reliable file sharing among friends.
- Building a novel weighted transaction network that enables file sharing among non-friends and resists free-riding, whitewash, collusion and Sybil attacks in P2P file sharing systems.

The remainder of this paper is arranged as follows. Section II presents the related work. Section III introduces the design of SocialTrust. Section IV presents the performance evaluation with real trace based simulation. Section V concludes this paper with future work.

II. Related Work

A. Reputation Management Systems

Reputation management systems in P2P networks have been widely studied [4]–[9]. EigenTrust [4] minimizes the influence of malicious nodes in a P2P network by calculating the global reputation of a node in the system as the left principal eigenvector of a matrix of normalized local reputation values. PowerTrust [5] uses a trust overlay network to model the trust relationships among nodes. It selects a few highly reputed "power nodes" to enhance global reputation accuracy and aggregation speed based on the discovered power-law distribution in user feedbacks from real traces. GossipTrust [6] leverages a gossip-based protocol to aggregate global reputation scores, which enjoys simplicity and moderate overhead. In [7], a distributed reputation management system for P2P system is proposed, in which the past behavior of a peer is reflected by its digital reputation with a novel cryptographic protocol. In [8], peers can either upload or download files at one time. Each node earns reputation by uploading files to others, and a server considers a client's reputation to decide whether to satisfy the client's request. Therefore, nodes are encouraged to contribute their network links for file uploading. BP-P2P [9] is a Belief Propagation (BP)-based distributed reputation management system for P2P networks. It selects a factor graph that can appropriately represent the P2P network to evaluate the reputation and trustworthiness of a node.

However, most previously proposed schemes are vulnerable to free-riding, whitewashing, collusion and Sybil attacks due to the dependence on the reputation values initialized and calculated based on feedbacks.

B. Social Network Based P2P File Sharing Systems

Several properties have been found on user-interaction based social network graphs such as "friendship fosters cooperation" [22], "average network properties remain relatively stable" [23], and "online social networks reflect those in the offline world" [24]. These properties have been exploited to support reliable services in P2P networks [13]–[21].

Turtle [13] constructs an overlay over the preexisting trust relationship between users to enable private and secure sharing of sensitive information through "friend-to-friend" file exchange. Tribler [14], an extension of BitTorrent [25], utilizes social phenomena such as friendship and the existence of communities of users with similar tastes to increase the usability and performance of a P2P network. F2F [15] is a cooperative data backup system, in which nodes select backup neighbors based on existing social relationships since friends tend to behave cooperatively on providing storage services to each other. MyNet [16] is a P2P platform that allows participating users to safely use and share their devices, services, and file resources with others without contacting any central control systems. In [17], social network links are utilized to conduct packet routing, thereby protecting peer privacy against Sybil attacks at a low complexity. SocialHelpers [18] uses the social network for node reputation evaluation in P2P systems. Each node collects the recommendations for a server from its trustworthy friends to determine the server's reputation. Social-P2P [19] groups common-multi-interest nodes into a cluster and connects socially close nodes within a cluster to achieve efficient and trustworthy file sharing. SocialTrust [20] also utilizes social network for efficient trust management in P2P file sharing systems. However, it uses reputation value to represent the trust among non-friends, which suffers from free-riding, whitewashing, collusion and Sybil attacks. Bartercast [21] exploits each node's local reputation graph and maxflow algorithm to detect free-riders in P2P networks.

However, these approaches either provide trustable transactions merely between friends or focus on solving a specific security concern such as free-riding. An effective mechanism is needed to guarantee trustable and fair file sharing among all nodes in P2P networks.

III. System Design

SocialLink provides efficient and trustable P2P file sharing service and meanwhile resists free-riding, whitewashing, and collusion/Sybil attacks. It incorporates two components: social network based server selection and weighted transaction network. In the social network, a node's social friends include both real-world social acquaintances and trustable on-line friends that always share files. When a client needs to select a server from server candidates, it selects its friends from the social networks, if available. When no friends are available in the server candidates, SocialLink relies on the weighted transaction network to decide whether the transaction with each server candidate is trustable and fair.

A. P2P Service Center

We assume a trusted P2P service center in the network that offers the P2P file sharing service with SocialLink enabled. It contains regular P2P file sharing functions (e.g., file indexing) and takes SocialLink as the trust management module. Nodes need to register at the service center to participate in P2P file sharing. When nodes request for a file, they send the request to the service center, which then returns the list of available servers following the server selection procedure in SocialLink (Sections III-B and III-C). After a transaction, nodes send feedback to the service center to update the transaction weight network (Section III-C). Since this paper focuses on trust management, we use SocialLink to represent P2P service center in this paper.

B. Social Network Based Server Selection

Generally, a user's friends in an online social network include both off-line social connections (e.g., relatives, friends, colleagues) and online connections. Similarly, the friends of a node in SocialLink also include both off-line acquaintances and trustworthy online friends that frequently share files with the node. When a node joins in the system, it is notified that only trustable nodes, either off-line acquaintances or online friends, can be added to its friend list. Hence, the friendship addition/deletion in SocialLink is user dependent behavior, and users are responsible for the consequence of adding/deleting a friend. Below, we first introduce how a node creates and maintains its social network and then present how friendships are used for server selection.

1) Social Network Construction: Each user creates and maintains its own friends in SocialLink. When a node (say N_i) wants to add another node (say N_j) into its friend list, it sends an invitation to N_j directly. N_j then decides whether to accept the invitation. The friendship is bidirectional in SocialLink. When N_i deletes N_j from its friend list, N_i is automatically removed from N_j's friend list. When a node joins in the system, it only adds off-line acquaintances as its friends. Later, the node adds online friends after it has conducted enough file transactions with other nodes. Specifically, when N_i has successfully downloaded a file from N_j, if there is no link connecting the two nodes, a weighted link is established from N_j to N_i; if there is already a link, its weight (denoted by w_{ji}) is updated based on the size of the shared file and the rating from N_i (the details are in Section III-E). When both w_{ji} and w_{ij} reach a predefined threshold T_f, SocialLink notifies N_i and N_j that they can be a friend of each other. Only when both nodes agree to add the other node as a friend, a friendship is established between them.

The social friendship in SocialLink represents a certain level of trustworthiness. Real-life social friends usually offer high quality-of-service (QoS) to each other. Users do not wish to damage their real-life reputations in their social communities (e.g., research lab or de-

partment) as a result of their misbehavior on online file sharing. Thus, the real-world friendship network motivates nodes to be cooperative continuously. Further, frequently interacted nodes of a node also have high probabilities to offer high QoS to the node according to their previous cooperative behaviors. Therefore, in order to maintain the friendship, which is a reflection of its trustworthiness, a node would not arbitrarily decrease the quality of services it provides to its friends.

2) Server Selection: SocialLink exploits friendship to realize fast server selection that can reduce the reputation querying cost. The general principle is that when requesting a file, a client asks for it from its friend directly, if available. Thus, the file sharing efficiency is improved by saving reputation querying. Specifically, when a client wants to download a file, it first finds the servers based on the P2P file searching algorithm. Then, the client checks whether there are any friends in the server list. If yes, the node selects a friend as the server for the file directly without querying the reputation values of all file owners. If there are multiple friends in the list, the node chooses the one with the highest trust. If there are no friends in the server list, the node uses the weighted transaction network to select a server, which will be introduced in Section III-C. After this file transaction, the client sends the feedback (positive, neutral, negative) to the system for link weight update.

C. Weighted Transaction Network

Recall that SocialLink regards the friendship as a representation of trust and enables friends to share files directly. However, a node usually has a limited number of friends. Therefore, solely relying on friendship for file sharing may greatly limit the amount of available file resources. Hence, it is necessary to provide reputation information for non-friends to enable freely and trustable file service. Besides, the fairness of file sharing among non-friends should also be considered to prevent free-riders. To realize the two objectives, SocialLink builds a weighted transaction network based on historical transaction records between nodes.

Figure 1 shows an example of part of the weighted transaction network in SocialLink. Nodes are connected by weighted directional links. When the system is started, since there are no transactions among nodes, the reputation information among non-friends cannot be evaluated. As friendship represents certain trust between friends, the weighted transaction network is initialized by creating links with weight $\triangle w$ connecting each pair of friends in both directions. The default link weight is to enable non-friends to be able to satisfy the requirements on file sharing in the beginning stage of the system, as introduced in Section III-D. Therefore, to be able to participate in the SocialLink network, a node must build friendships when it joins in. Otherwise, it has no chances for file sharing and will be isolated from the network. $\triangle w$ is usually set to a medium value to ensure the bootstrap of the system. SocialLink allows downloading a large file from several servers to ensure that large files can be downloaded in the bootstrap stage.

As more file transactions occur in the system, weighted links are constructed between non-friends. When a node (say N_i) successfully provides a file to another node (say N_j), the weights of links on the path used to evaluate the trustworthiness and fairness of the transaction are updated according to the file size and the rating on the file quality from the client. Also, if there is no link from N_i to N_j, a link is built from N_i to N_j with weight being equal to the file size. The link weight means the size of files that N_j can trustingly obtain from non-friend N_i. The details of the usage of the weighted transaction network will be introduced in Section III-D, and the link weight update will be introduced in Section III-E.

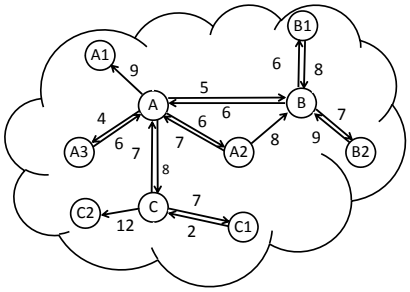


Fig. 1: A part of the weighted transaction network in SocialLink.

D. File Sharing Process in SocialLink

When a client requests a file, if there are friends in the candidate server list, the client directly asks the file from a selected friend, as introduced in Section III-B2. Otherwise, the weighted transaction network is used to decide whether to execute the transaction and which server to be selected for the request.

Recall that the weight of a link $N_i \rightarrow N_j$ represents the size of files that N_j can trustingly obtain from N_i based on its previous file provision records. It also means that N_j should upload files with size no smaller than this weight to N_i for fair trading and free-riding prevention. For example, the link of $B \rightarrow A$ in Figure 1 means that A believes that it can obtain 6MB trustable files from B. Also, A should upload files with size no smaller than 6MB to B for fair trading.

Two nodes can be connected by a number of links, which form a path (i.e., $N_i \rightarrow \cdots N_j$). We regard the smallest link weight in all the links on a path as the weight of the path, which represents the trust of N_j on N_i in providing the file based on the trust transited along this path. There may be multiple paths

connecting N_i to N_j. We then define the *trust-flow* as the largest path weight among the weights of all paths from a server candidate (N_i) to the client (N_j) (denoted by W_{ij}). It represents the client node's trust on the server candidate's trust to provide files within size W_{ij}. Similarly, we define the *upload-flow* as the largest path weight among the weights of all paths from the client to the server candidate (i.e., $N_j \rightarrow \cdots N_i$). It represents the accumulated size of files that the server candidate (N_i) should provide to the client (N_j) for fair trading. Then, the trust-flow is used to evaluate whether a server candidate is trustable in providing the requested file, and the difference between the trust-flow and the upload-flow is used to ensure the fairness of this file transaction. The "six degrees of separation" [26] property of social networks indicates that two people are connected by a maximum of six steps in the world on average. Thus, we limit the server-client and client-path path length in the weighted transaction network to 6 hops. Paths with length longer than 6 are not considered in SocialLink.

In detail, suppose a client node (say N_i) requests a file with size S. If the server candidate list includes k servers $N_{s1}, N_{s2}, \cdots, N_{sk}$ and none of them are N_i's friends, the file sharing process follows below.

- SocialLink calculates the client's upload-flows to each server, denoted by UF_{im}, and each server's trust-flow to the client, denoted by TF_{mi}, where $m \in [1, k]$.
- To ensure fairness, only servers with $(UF_{im} - TF_{mi}) \geq S$ or $|UF_{im} - TF_{mi}| \leq Th_r$ are willing to provide files to N_i, where Th_r is a fairness threshold. The first condition means that the server owes more than the size of the requested file to the client to make their trading fair and the second condition means the gap on mutual contribution is on a roughly equal status, which means transactions can be allowed between the two nodes. Then, only these servers are kept in the server list.
- For trustable file sharing, servers with trust-flow smaller than the requested file's size S are removed from the list since they are not trustable to provide the file based on their historical records.
- Finally, the client chooses the server with the highest trust-flow for the file transaction because it is the most trustable one in terms of provided file size.

In this way, SocialLink ensures that the server has a high probability to provide the requested file and the client is not a free-rider. It is possible that the server candidate list might be empty after the first three steps. In this case, to support broad file sharing services, SocialLink still returns several servers with high TF_{mi} and high $(UF_{im} - TF_{mi})$ or low $|UF_{im} - TF_{mi}|$ as a backup strategy. This is because these servers are close to satisfy the requirement for server candidates and thereby are assumed to be very likely to be trustable for the requested file service. The client can consider other factors such as social connection strength, common file interests, or third party reputation system to decide whether to conduct the transaction with one of these servers.

We use an example based on Figure 1 to explain the process of a file transaction. When $C1$ sends a file request for a file with size 4MB and $B2$ owns the file, $C1$ is the client and $B2$ is a server candidate. Because $B2$ is not a friend of $C1$, $C1$ queries SocialLink to check whether the transaction with $B2$ will be malicious. Then, SocialLink examines links from B2 to C1 and finds that the trust-flow from B2 to C1 is 6 through the path $B2 \rightarrow B \rightarrow A \rightarrow C \rightarrow C1$. Similarly, the upload-flow from C1 to B2 is 2 through the path $C1 \rightarrow C \rightarrow A \rightarrow B \rightarrow B2$. Since the difference between the trust-flow and the upload-flow, i.e., the difference of mutual trust on providing files, equals to the file size (i.e., 4), and the trust-flow is larger than the file size (i.e., $6 > 4$), SocialLink knows that the B2 can provide the file to C1 with high quality. Such information is then sent to both C1 and B2 to approve the transaction.

E. Link Weight Update in SocialLink

Link weight is not updated after a transaction between two friends. This is because the link weight is designed to provide an evaluation for the trust among non-friends, and a node's QoS to its friends cannot reflect its QoS to non-friends. Thus, updating the link weight between two friends will make the evaluation of trust between non-friends inaccurate. For example, in an extreme case, two friends always provide good services and feedbacks to each other and bad services to non-friends. If the transaction records between them are considered in calculating trust-flow and upload-flow, it will mislead other nodes on the trust of the two nodes.

The link weight is updated only after a transaction between two non-friends. In this case, the client reports the service rating to SocialLink: positive, neutral, and negative. Based on the rating, SocialLink updates link weights as in the following.

• **Positive feedback** After a client receives a satisfactory file, it provides positive feedback to the server. Then, the weight of each link along the path for the trust-flow is added by the size of the file. If the server and the client are not directly connected, a new link is established between them with its weight equals to the file size.

• **Neutral or No feedback** If the client reports a neutral feedback or does not provide a feedback, SocialLink does not update the link weights. A neutral feedback or the absent of the feedback means that the file from the server is not faulty, but the client may not be completely satisfied with the file (e.g., medium quality). In this case,

it is not necessary to punish the server, and the client may want to keep the link to the server for future transactions.

• **Negative feedback** When the non-malicious client provides a negative feedback for faulty files received from the server, SocialLink lowers the weight of each link on the path used to calculate the trust-flow by the size of the transferred file. Links with weights smaller than 0 are removed directly.

In addition to above design, we also fade weight links over time and set a limit for link weight for practical consideration. First, the weight of a link is faded with a factor $\beta \in [0.5, 1]$ every T_d. The fading factor makes recent behaviors more important in deciding a node's trust. The value of β and T_d should be determined by how active nodes are in the system. We usually set β and T_d to a large value, i.e., 0.95 and 1 day, respectively, to avoid disconnecting the network. Further, for practical consideration, we also set a maximal weight for a link to prevent it from overflow, which is set as the maximal value that can be represented by the link weight variable in the actual system (i.e., 255 for 8 bit integer).

Such a design encourages both server and client to be cooperative. First, if a node receives negative feedbacks frequently, all of its links to other nodes will be removed eventually, and it will have few opportunities to obtain files from other nodes or provide files to others to rebuild its links. As a result, server nodes are encouraged to provide high-QoS files. Second, when a client provides a negative or neutral feedbacks for a high quality service, the weight of the path from the high-QoS server to the node is reduced or remains unchanged. Then, high-QoS servers cannot be distinguished from others, which may prevent the client node from finding trustable servers in the future. On the other side, when a client provides a positive feedback to a low-quality file, it increases the trust of the misbehaving server. This means that it is likely to receive low-quality files again. In summary, nodes are encouraged to provide correct feedbacks for received files to ensure their benefits in SocialLink.

F. Understanding the Meaning of the Link Weight

In SocialLink, the weight of a link between two non-friends, say the link from N_i to N_j (denoted by L_{ij}), serves as N_j's trust on N_i's ability to provide files to others. However, two nodes may not always be connected with a link. They may be connected by a path with multiple links, as shown in Figure 1. In this case, we adopt the concept of trust relaying to calculate the trust on file providing ability represented by the path. Specifically, suppose there is a path with 3 links: $N_a \rightarrow N_b \rightarrow N_c \rightarrow N_d$, and their weights are $w_{ab} = 5$, $w_{bc} = 3$, $w_{cd} = 7$, respectively. Then, N_b recommends N_c that N_a's file providing ability is w_{ab}. However, since a node's recommendation should be limited by its trust, we adopt the smaller one of its trust and its recommendation as its effective recommendation. Therefore, N_c's trust on N_a's file providing ability is $w_{bc} = 3$ since $w_{bc} < w_{ab}$. Similarly, since $w_{cd} > w_{bc}$, N_d's trust on N_a's file providing ability is still $w_{bc} = 3$. In summary, we use the smallest link weight on the path to represent the ending node's (N_d's) trust on the starting node's (N_a's) ability to provide files.

Consequently, as mentioned in Section III-E, L_{ij} is updated in two cases. First, after N_i provides a file to N_j for the first time (L_{ij} is not created yet), L_{ij} is initialized with its weight equals to the size of the shared file. Second, when a path that includes L_{ij} has been used to represent the trust in a transaction, the weight of L_{ij} is updated based on the rating for the transaction. In summary, the weight of L_{ij} is updated accordingly to reflect the change of either the direct trust or the recommendation represented by the link.

With such a design, the weight of a link can be changed by transactions conducted between other nodes. As a result, malicious nodes can purposely lower the weights of the links of a node to disconnect the node or reduce the weights of some paths. We discuss how to prevent such an attack in Section III-H4. However, such a feature actually protects the system somehow. The former case means that the node is surrounded by malicious nodes. Then, the decrease of the weights of its links reflect the true situation, i.e., its recommendation is very low. The latter case prevents nodes from adopting the low weight paths. Finally, the probabilities that malicious nodes are selected as servers are reduced, which benefits the file sharing system.

G. Computing and Communication Complexity.

SocialLink has a single trust manager (i.e., P2P service center) to find paths from a client to a file server in the weighted transaction network. For each file request, the communication overhead is $O(1)$ between a client and the P2P service center. To find the paths from a client to a file server, SocialLink depends on a breadth-first search solution. Therefore the computing complexity is $O(m + n)$, where m and n are the number of edges and nodes in the weighted transaction network, respectively. Such complexity is acceptable for modern computing servers.

H. Attack Resistance and Extensions

We further briefly discuss how SocialLink works under several malicious behaviors and propose possible extensions to better prevent these attacks.

1) Free-riding: The consideration for fairness during file transaction, as introduced in Section III-D, prevents the free-riding in SocialLink. A server candidate is willing to provide a file to a client when the difference between upload-flow and trust-flow satisfies either

$(UF_{im} - TF_{mi}) \geq S$ or $|UF_{im} - TF_{mi}| \leq Th_r$, which mean that the client node has provided at least S more files than the server has provided or the server and the client have provided similar amount of files to each other. Then, when a node is reluctant to contribute to others, other non-friends are not willing to provide files to it too. Consequently, free-riding is prevented in SocialLink.

2) Whitewashing: Whitewashing cannot enable a malicious node to continue providing faulty files or be a free-rider in SocialLink. First, in the weighted transaction network, a link is created only after a successful transaction. When a whitewasher creates a new account, it only has links to friends (Section III-C). Then, it can only rely on friends' links to other nodes to be selected by non-friends as the file server. However, a whitewasher can hardly always have normal friends unless they are colluding, the prevention of which will be discussed in Section III-H3. As a result, without links, whitewashers will not be selected by non-friends as servers and thereby cannot send faulty files to non-friends.

Similarly, a whitewasher can only download files through its friends' links to other nodes. As a result, they can hardly continuously free-ride due to the same reason mentioned above. Even when they have a few friends, the fairness consideration (Section III-H1) will soon forbid them to download files from others. Consequently, nodes can hardly have free-riding in SocialLink.

3) Collusion and Sybil Attack: In collusion and Sybil attack, colluding nodes purposely increase the weights of links connecting them in the weighted transaction network. They wish to use such weight-boosted links to "deceive" other nodes that they are trustable. SocialLink can resist such malicious behaviors to a certain degree naturally. When colluding nodes provide faulty files to non-malicious nodes outside the collusion group, the weights of all links in the path for the trust-flow path are reduced. Then, though colluding nodes have high-weight links connecting each other, the weights of their links to outside nodes are very low or even 0. Since the weight of a path is determined by the smallest weight of its links, outside nodes will have a low trust on these colluding nodes and will not choose them as the file provider. As a result, colluding nodes can hardly harm outside nodes due to the lack of high-weighted links with them.

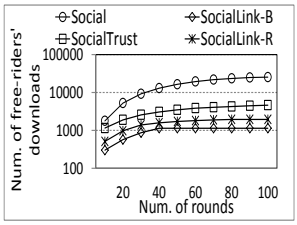
4) Bad-mouthing Attack: In the bad-mouthing attack, a group of nodes purposely provide negative feedbacks to a node. As a result, though the node is cooperative, it is isolated from the network. As introduced in Section III-E, the design of SocialLink can prevent a rational node from reporting manipulated feedbacks. However, this is not enough to thwart the bad-mouth attack.

There are already some bad-mouthing defense methods [20], [27] that can be incorporated to solve this problem. We adopt the scheme in our previous work [20] in this paper. Basically, the scheme allows a node to file a claim against incorrect feedbacks for it. Then the bad-mouthing attack becomes the he-said-she-said attack, and we just need a way to check which node is honest. Our method uses an intuitive way for this purpose by comparing the file against previous high quality files for the request. Automatic file quality detection algorithms can also be applied in this process. After the quality check, the dishonest node (i.e., filing a wrong claim or incorrect feedback) will be punished heavily. Please refer to [20] for more details about such a solution.

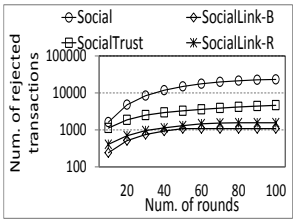
IV. Performance Evaluation

We conducted extensive simulation to show the performance of SocialLink in saving querying cost, detecting malicious transactions, preventing malicious behaviors in different network scales. We randomly selected a medium social network with 5,000 nodes from LiveJournal [28], which is a social network trace, to build their mutual friendship in our simulation. We define three types of nodes: good, neutral and bad. In order not to be biased, we randomly selected bad nodes, i.e., malicious nodes, among all nodes. Due to the small scale of the number of nodes, in order not to make the weighted transaction network disconnected after blocking suspicious transactions, we selected 10% nodes as bad nodes. Bad nodes provide low-quality files, and always give dishonest feedbacks, such as negative feedbacks to non-malicious nodes. We randomly selected 70% and 20% of nodes as good nodes and neutral nodes, respectively, i.e., non-malicious nodes. Good nodes and neutral nodes provide high-quality files and medium-quality files, respectively, and all non-malicious nodes give honest feedbacks. We simulated one single trust manager to store and respond the request for the weighted transaction network. We assume the centralized manager having enough storage and computing capacity to serve all requests regarding weighted transaction network.

In order to measure the performance of each node in the simulated social network from the trace without bias, we assumed a random distribution for all parameters in the settings. We configured 1,000 files with sizes randomly selected from [1,100] MB. Each file has n file replicas, where n is randomly chosen from $[1, 5]$. We also define the quality of files by levels randomly selected from range [1,10] to distinguish different types of files. Good files have quality levels randomly selected from range [7,10], neutral files have quality levels randomly selected from range [4,6] and bad files have quality level in range [1,3]. The file holder for a high-quality, medium-quality and low-quality file replica is randomly selected from good, neutral and bad nodes, respectively. In our experiments, when the size of the P2P file sharing system is enlarged by m times, the number of the files

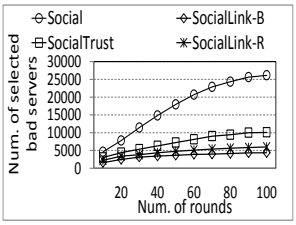


(a) Accumulated number of free-riders' downloads.

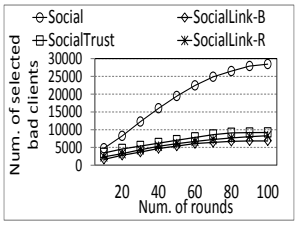


(b) Accumulated number of rejected transactions.

Fig. 2: Accumulated number of rejected requests and free-riders' downloads.



(a) Accumulated number of selected bad servers.



(b) Accumulated number of selected bad clients.

Fig. 3: Performance with the whitewash misbehavior.

is also enlarged by m times with the same percentages of good, neutral and bad nodes. We set Th_r to be the largest file size in order to make sure there is at least one file that can be shared between the server and client at the beginning. We set T_f as two times of the largest file size. It means strangers become friends after sharing 4 files on average in an experiment.

Friendship threshold T_f is set to 200 in our experiments. We ran each experiment for consecutive 100 rounds. In each round, every node generates a file request. The requested files of a node were randomly selected from the files not owned by the node. Based on the quality of the received file and the type of the client, different feedbacks were given to the server. A node can share its received files with other nodes later.

We compare SocialLink with a reputation system based on social trust, denoted as *Social*, in which each node defines the maximum path length to search for requested files. According to [29], people are influenced by other people who are at most at 3-hop social distance, and due to our small community, we set the maximum path length as 2 in *Social*. We also compare SocialLink with *SocialTrust* [20], a social network based reputation management system. In *SocialTrust*, the reputation value of each node is initialized to 0 and increases/decreases by 1 upon receiving a positive/negative feedback. Social network is also built based on interactions in *SocialTrust* to facilitate server selection. The threshold of being selected as servers is set to 0 by default. Recall that in SocialLink, suspicious transactions can be blocked or a reputation management system can be used to select the server with the highest reputation value. We use *SocialLink-B* and *SocialLink-R* to denote SocialLink with these two strategies, respectively, and tested their performance in experiments.

A. Accuracy in Detecting Suspicious Transactions

We measured the number of total suspicious transactions and the number of falsely marked transactions in each round. We regarded the falsely marked suspicious transactions as false negative and falsely marked normal transactions as false positive. Hence, the percentage of falsely marked suspicious transactions is a method to measure the accuracy of *SocialLink-B* in detection fraudulent transactions.

# rounds	10	20	30	40	50	60
# suspicious trans.	2694	826	187	60	15	10
# false negative	2454	712	115	32	4	1
% of false negative	91.1	86.3	61.3	53.2	26.7	10
# malicious trans.	212	208	182	153	87	24
# false positive	9.1	5.0	3.9	3.1	1.8	0.5

TABLE I: Percentage of falsely marked transactions by *SocialLink-B*.

From Table I, we see that the number of falsely marked suspicious/normal transactions is large in the beginning, and then gradually decreases in the subsequent rounds. This is because when a new weighted transaction network is established, the links between nodes are not fully built due to the absence of transaction history. Also, the paths of friend-of-friend connection may be used to select bad nodes as servers. Hence, a high percentage of transactions are detected as suspicious, and there are 91.1% and 9.1% for false negative and positive, respectively. After directional weighted links are generated between nodes as the number of transactions increases, the number of suspicious transactions decreases rapidly, so is the number of falsely marked transactions.

B. Preventing Free-riding

In free-riding, nodes tend to reject requests but download freely from non-friend nodes. In this test, we assumed that 20% of 5,000 nodes are free-riders in the system that has 50% probability to reject file requests.

Figure 2(a) illustrates the accumulated number of free-riders' downloads throughout the test. We observe that the result follows *SocialLink-B*<*SocialLink-R*<*SocialTrust*. This is because in SocialLink, free-riders do not contribute downloading to others. Thus, transactions are blocked for not finding a reliable path from the free-rider to the server with $(UF_{im} - TF_{mi}) \geq S$ or $|UF_{im} - TF_{mi}| \leq Th_r$. On the contrary, *SocialTrust* cannot prevent all free-riders with high reputation values from downloading files from non-friend nodes since it fails to consider fairness. *SocialLink-R* checks the link path between nodes in the weighted transaction network when it exists. It manages suspicious transactions by querying the reputation values of available servers. As

a result, some free-riders still can request files when they have high reputation values. Therefore, it generates fewer downloads from free-riders than *SocialTrust* and more rejected transactions than *SocialLink-B*. *Social* has the most downloads from free-riders since it does not use reputations to distinguish malicious nodes with close social relationship. Thus, it cannot prevent free-riders.

Figure 2(b) presents the accumulated number of rejected transactions. The results follow *SocialLink-B*<*SocialLink-R*<*SocialTrust*<*Social* due to the same reason as Figure 2(b). Therefore, the accumulated number of rejected transactions does not increase in the subsequent rounds in *SocialLink-B*. This figure indicates that SocialLink successfully prevents free-riders to be selected as servers. Both Figures 2(b) and 2(a) indicate that SocialLink can protect users from free-riders.

C. Reducing the Adverse Effect of Whitewash

This experiment tests how SocialLink reduces the adverse effect of whitewash. In this test, in each round, 50% of all malicious nodes deleted their current accounts and created new ones with friendship initially. By whitewashing, they remove low reputation values and restore their reputation values to initial value 0 after whitewashing. The number of selected bad servers and clients refer to the number of transactions that take bad nodes as servers and clients, respectively.

Figure 3(a) shows the accumulated number of selected bad servers of all systems, which follows *SocialLink-B*<*SocialLink-R*<*SocialTrust*<*Social*. *Social* selects the largest number of bad servers due to the same reason as Figure 2(a). *SocialTrust* can prevent malicious behavior of bad servers, but with the whitewash, malicious nodes clean their low reputation values and are selected as servers again in the subsequent rounds. Thus, it selects fewer bad servers than *Social*, but more bad servers than the other methods. However *SocialLink-B* removes all links of a node when the account of the node is deleted. Even though bad nodes connect with their friends immediately, there is no link to non-friend nodes because no transaction is conducted. Hence, no malicious nodes are selected as servers with the whitewash misbehavior while all links are established as the experiment goes on. *SocialLink-R* conducts suspicious transactions with reputation values, so that some bad nodes are selected as servers when reputation values are not accurate enough. Then, with the weighted transaction network, only a small number of bad nodes are selected as servers due to the same reason as SocialLink. This figure indicates that SocialLink can prevent the malicious servers from Whitewash better than the others.

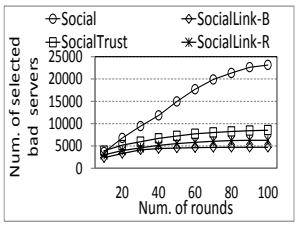
Figure 3(b) presents the accumulated number of selected bad clients. We see that the results follow *SocialLink-B*<*SocialLink-R*<*SocialTrust*<*Social* due to the same reasons as in Figure 3(a). This figure indicates that SocialLink prevents the Whitewashed attack better than other systems.

D. Resisting Collusion and Sybil Attacks

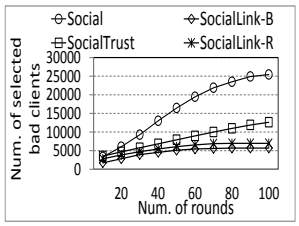
In this test, we assume that all bad nodes colluded with their colluding nodes or Sybils a long time before the 1^{st} round to measure the damage of these threats. With this assumption, each bad node in *SocialLink-B*, *SocialTrust*, *SocialLink-R* and *Social* conducts 100 transactions with randomly selected colluders and receives all positive feedbacks. Even though nodes collude in the same way in all systems, the results of collusion are different. Bad nodes in SocialLink build links with expected weight values as 5000, since the expected file size is 50, but they still have no links to non-friend nodes since no transactions happened between non-friend nodes. On the other hand, bad nodes in *SocialTrust* increase their reputation values to 100 before the 1^{st} round.

Figure 4(a) shows the accumulated number of selected bad servers by non-friends, which indicates the accumulated number of low-QoS transactions received by nodes. We find that the results follow *SocialLink-B*<*SocialLink-R*<*SocialTrust*<*Social* in all rounds. In *SocialLink-B*, the malicious colluders' entire community will be blocked by others, since the weights of links connecting colluders to others decrease by sending bad files to others outside this community. Thus, it generates the smallest number of transactions with bad nodes as servers. On the other hand, *SocialTrust* selects bad nodes as servers that gain high reputation values by colluding. Only after many transactions to correct the reputation values of all nodes, *SocialTrust* can distinguish bad nodes and stop selecting them as servers. However, the colluders can make conclusion again to defeat *SocialTrust*. Instead of blocking suspicious transactions, *SocialLink-R* uses reputation values to conduct suspicious transactions. Since some bad nodes cannot be distinguished with high reputation values by colluding, the accumulated number of selected bad servers in *SocialLink-R* is more than *SocialLink-B* and less than *SocialTrust*. As a result, the *SocialLink-R* selects fewer bad nodes as servers based on the weighted transaction network. *Social* selected the largest number of bad servers; since it cannot distinguish bad nodes by only trusting social close nodes within 2 hops. This figure indicates that SocialLink can successfully prevent the colluded servers to supply bad files.

Figure 4(b) shows the accumulated number of selected bad clients of all methods, which follows *SocialLink-B*<*SocialLink-R*<*SocialTrust* due to the same reason as Figure 4(a). This figure indicates that SocialLink can successfully defeat the colluded clients.

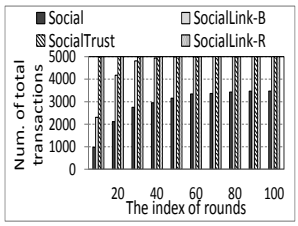


(a) Accumulated number of selected bad servers by non-friends.

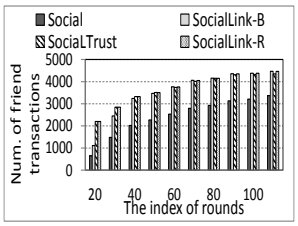


(b) Accumulated number of selected bad clients by non-friends.

Fig. 4: Performance with the collusion and Sybil attacks.



(a) Number of total transactions.



(b) Number of friend transactions.

Fig. 5: Number of total and friend transactions.

V. Conclusion

In this paper, we propose a social network and transaction links based reputation system, namely SocialLink, to provide efficient and effective trust management in for P2P file sharing systems. SocialLink exploits the social network to allow friends to share files directly and efficiently. In order to enable file sharing among non-friends, SocialLink designs a novel weighted transaction network to manage the trust among non-friends based on transaction records, in which the weight of a link from a node to another node denotes the size of files that the latter can trustingly obtain from the former. Such a design prevents certain attacks such as whitewashing, collusion and Sybil attacks since malicious nodes cannot interact with other nodes without links connecting them, i.e., without providing good files to them. Finally, extensive real trace based experiments demonstrate the efficiency and effectiveness of SocialLink. In the future, we plan to more accurately model the weighted transaction network.

Acknowledgements

This research was supported in part by U.S. NSF grants NSF-1404981, IIS-1354123, CNS-1254006, CNS-1249603, Microsoft Research Faculty Fellowship 8300751. We would like to thank our shepherd, Dr. Bernard Wong. His insights and suggestions have helped us improve the quality of the paper significantly.

References

[1] E. Adar and B. A. Huberman, "Free riding on gnutella." *Technical report, Xerox PARC*, 2000.

[2] D. Hughes, G. Coulson, and J. Walkerdine, "Free Riding on Gnutella Revisited: The Bell Tolls?" *IEEE Dist. Systems Online*, 2005.

[3] S. Shin, J. Jung, and H. Balakrishnan, "Malware prevalence in the kazaa file-sharing network." in *Proc. of IMC*, 2006.

[4] S. Kamvar, M. Schlosser, and H. Garcia-Molina, "The EigenTrust algorithm for reputation management in P2P networks," in *Proc. of WWW*, 2003.

[5] R. Zhou and K. Hwang, "PowerTrust: A Robust and Scalable Reputation System for Trusted Peer-to-Peer Computing," *IEEE TPDS*, 2007.

[6] R. Zhou, K. Huang, and M. Cai, "GossipTrust for Fast Reputation Aggregation in Peer-To-Peer Networks," *IEEE TKDE*, 2008.

[7] P. Dewan and P. Dasgupta, "P2P reputation management using distributed identities and decentralized recommendation chains." *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 7, pp. 1000–1013, 2010.

[8] A. Satsiou and L. Tassiulas, "Reputation-based resource allocation in P2P systems of rational users." *IEEE Trans. Parallel Distrib. Syst.*, vol. 21, no. 4, pp. 466–479, 2010.

[9] E. Ayday and F. Fekri, "BP-P2P: Belief propagation-based trust and reputation management for P2P networks." in *Proc. of SECON*, 2012.

[10] "eBay," http://www.ebay.com.

[11] "Amazon.com," http://www.amazon.com.

[12] "Overstock," http://www.overstock.com/.

[13] B. Popescu, B. Crispo, and A. Tanenbaum, "Safe and Private Data Sharing With Turtle: Friends Team-Up And Beat The System," in *Proc. of SPW*, 2004.

[14] J. A. Pouwelse, P. Garbacki, J. Wang, A. Bakker, J. Yang, A. Iosup, D. H. J. Epema, M. Reinders, M. van Steen, and H. J. Sips, "Tribler: A social-based peer-to-peer system," in *Proc. of IPTPS*, 2006.

[15] J. Li and F. Dabek, "F2F: Reliable Storage in Open Networks," in *Proc. of IPTPS*, 2006.

[16] D. N. Kalofonos, Z. Antonious, F. D. Reynolds, M. Van-Kleek, J. Strauss, and P. Wisner, "MyNet: A Platform For Secure P2P Personal And Social Networking Services," in *Proc. of PerCom*, 2008.

[17] P. Mittal, M. Caesar, and N. Borisov, "X-Vine: Secure and pseudonymous routing using social networks," *CoRR*, 2011.

[18] M. S. Artigas and B. Herrera, "SocialHelpers: Introducing social trust to ameliorate churn in P2P reputation systems." in *Proc. of Peer-to-Peer Computing*, 2011.

[19] Z. Li and H. Shen, "Social-P2P: Social network-based P2P file sharing system," in *Proc. of Network Protocols (ICNP)*, 2012.

[20] K. Chen, H. Shen, K. Sapra, and G. Liu, "A social network integrated reputation system for cooperative P2P file sharing," *IEEE TPDS*, 2015.

[21] M. Meulpolder, J. A. Pouwelse, D. H. Epema, and H. J. Sips, "Bartercast: A practical approach to prevent lazy freeriding in p2p networks," in *Proc. of IPDPS*. IEEE, 2009, pp. 1–8.

[22] E. Pennisi, "How did Cooperative Behavior Evolve?" *Science*, 2005.

[23] B. Viswanath, A. Mislove, M. Cha, and K. P. Gummadi, "On the evolution of user interaction in facebook," in *Proc. of WOSN*. ACM, 2009, pp. 37–42.

[24] R. Dunbar, V. Arnaboldi, M. Conti, and A. Passarella, "The structure of online social networks mirrors those in the offline world," *Social Networks*, vol. 43, pp. 39–47, 2015.

[25] "BitTorrent," http://www.bittorrent.com/.

[26] M. Newman, A. Barabsi, and D. Watts, *The Structure and Dynamics of Networks*. Princeton University Press, 2006.

[27] Y. L. Sun, Z. Han, W. Yu, and K. R. Liu, "A trust evaluation framework in distributed networks: Vulnerability analysis and defense against attacks." in *INFOCOM*, vol. 2006, 2006.

[28] J. K. X. L. G. L. Backstrom, D. Huttenlocher, "Group Formation in Large Social Networks: Membership, Growth, and Evolution," in *Proc. of KDD*, 2006.

[29] N. A. Christakis and J. H. Fowler, *Connected: The surprising power of our social networks and how they shape our lives*. Hachette Digital, 2009.