International Journal of Security and Its Applications
Vol.8, No.4 (2014), pp.329-340
http://dx.doi.org/10.14257/ijsia.2014.8.4.30

Defending Against sybil-attacks in Peer-to-Peer Networks

Xu Xiang, Lu Huijuan and Chen Lianna

College of Information Engineering, China Jiliang University
xu@cjlu.edu.cn{hjlu,chenlianna}@cjlu.edu.cn

Abstract

Peer-to-Peer networks have become a popular way for users to share files over the Internet.However, there has been a spurt of works showing that the existence of sybil attacks is a serious threat to Peer-to-Peer networks, where one or more attackers can forge a large number of fictitious identities. Implementing correct protocols to address sybil attacks is the key to improving the performance.In this paper, we present a novel systemto defend against Sybil attacks.Our direct and indirect transaction protocols limit the number of service units that a node can obtain.Furthermore, we design a dynamic reputation ranking algorithm for the indirect transaction protocol.Combining these two, a node with a high priority has more probability of obtaining service.Our system does not try to prevent users from creating multiple identities, but they cannot gain extra profit from doing so.It achieves a provable performance and overcomes the limitation of current social network-based defenses.Simulation results show that our system achieves a provable bound in terms of the number of service units obtained by sybil nodes while not sacrificing the performance of the file sharing application.

Keywords: *Peer-to-Peer, Sybil attack, Free riding*

1. Introduction

Peer-to-Peer (P2P) networks have attracted a significant amount of interest because of their desirable features such as being scalable, decentralized and anonymous. Among their numerous applications, file sharing is the most important and popular function. In file sharing systems, files are stored at nodes according to pre-established rules and nodes automatically provide files for sharing with other nodes.

However, a high degree of *free riding*[1] has been observed in reality, which adversely affects the file sharing application. Some nodes may consume service units of file sharing from providers (downloading files), but be reluctant to provide service units (uploading files) to others in order to save their own resources. Resources and network capacities will be improperly occupied by those nodes, and therefore the function of file sharing may fail.

The existing approaches [2-4] based on reputation mechanisms have become a promising way to address the free riding problem. For each node, reputation systems calculate reputation scores according to its historical behavior, by which nodes may obtain a certain number of service units (downloading files).

Unfortunately, reputation systems are susceptible Sybil attack [5]. As a P2P concept, sybil attack means creating multiple identities by one or more adversaries, causing serious negative side effects. It is known to be a fundamental threat against the working of distributed systems including P2P networks. Sybil nodes may create arbitrary reputation values among fictitious identities, rendering reputation mechanism obsolete.

ISSN: 1738-9976 IJSIA

There is excitement in the research community about using social networks to identify sybil regions. These sybil defense schemes assume that sybil nodes cannot establish an arbitrarily large number of edges connecting to non-sybil regions. They leverage the feature to identify sybil regions with different graph analysis techniques.

Recent comparative research[6] provides a common insight that explains how all these schemes are able to detect sybil regions, showing that, despite their considerable differences, all social network-based sybil defense schemes rely on identifying communities in the social network. This work also suggests that it is possible to use off-the-shelf community detection algorithms to detect sybil regions. However, it also discovers that these schemes and algorithms relying on community detection make networks more vulnerable to sybil attacks.

In this paper, we focus on designing high level mechanisms that address sybil attacks in free riding of P2P networks. We propose protocols which:(1) can limit the number of service units that sybil nodes obtain, and therefore defeat sybil attacks.(2) can bring high efficiency to overall network because of fewer rejections in transactions between cooperative nodes.(3) do not leverage topological features to identify communities, and therefore overcome the inherent limitations of current social network-based schemes. Our protocol is fully distributed and applicable to both structured and unstructured P2P networks. Nodes are only required to know their neighbors and take actions based on local information.

The rest of this paper is organized as follows. In Section 2, we present related work. Section 3 describes our system model. We design sybil-resilient protocols in Section 4. Section 5 defines a dynamic reputation ranking algorithm. Section 6 presents simulation results for some network model. Conclusions are given in Section 7.

2. Related Work

Relying on a central authority certifying identities to prevent sybil attacks was proposed in [1]. Several works [7-8] follow the idea and present similar schemes. A main disadvantage is that a central authority causes a single point of failure that results in a scalability problem. Moreover, the central authority becomes a potential target for attacks.

Verifying resources of each node, such as CPU capability, network bandwidth and IP address, is based on the assumption that attackers can only possess limited resources. Each node should face challenges, and prove that it actually occupies designated resources. Otherwise, it will be identified as a sybil node and does not take part in P2P networks. Several works [9-10] use cryptographic puzzles to detect possible sybil nodes. Baumgart, *et al.*, [11] limits the generation of node IDs using crypto puzzles. Those solutions only limit the rate with which the attacker can introduce sybil nodes into P2P network, rather than the number of introduced sybil nodes.

Sybilproof [12] converts a P2P network into a trust network to defend against sybil attack. But it aims to prevent the attacker from boosting reputation values among sybil nodes. It focuses on a static graph model of reputation, instead of a dynamic reputation model. In this paper, we develop a dynamic reputation ranking algorithm.

The known promising ways for defending against sybil attacks are recent works in node admission control applying social networks. Unlike traditional ways, those protocols require no control centers to identity nodes. They make the assumption that although sybil nodes can forge an arbitrary number of identities, they have no power to establish a large number of links connecting to non-sybil nodes. The interconnections between sybil nodes and non-sybil nodes are rather sparse. They leverage this topological feature of social networks to detect sybil regions.

SybilInfer [16] is a centralized protocol which uses Bayesian inference to assign each node a probability of being a sybil node. SumUP [17] is a centralized admission protocol which admits nodes computing max-flow paths from a vote envelope to all nodes. Gatekeeper [18] further improves over SybilLimit by a factor of $O(\log n)$ on random expander graphs when the attacker dominates only $O(1)$ attack edges. The above three protocols implicitly or explicitly make similar assumptions as SybilGuard and SybilLimit.

SybilGuard [13] is the first known admission control protocol in social networks. Instead of a third-party authority to establish trust relationships among nodes, it uses a distributed verification protocol based on random routes, limiting the number of admitted sybil nodes to $O(\sqrt{n}\log n)$ per attack edge, where n is the number of honest nodes in the social network. SybilLimit [14] improves this bound and reduces the number of Sybils admitted per attack edge to $O(\log n)$ with high probability. Both of them are designed to work in a distributed system where each node is initially only aware of its neighbors. Similarly, Sybil Defender [15] also leverages the network topologies to defend against sybil attacks in social network.

But there is a question whether or not those protocols used in social networks can be implemented in P2P networks? Unlike social networks, nodes in P2P networks are anonymous and hence do not know each other. Recent work [6] makes a comparative analysis of social network-basedsybil defenses including SybilGuard, SybilLimit, SybilInfer and SumUp over a given set of social networks. It shows that each protocol contains an algorithm which produces a ranking of nodes, and works by implicitly ranking nodes based on how well the nodes are connected to given trusted nodes. The study reveals the limitation of relying on community structure of the social network to distinguish sybil nodes from non-sybil node. Relying on community detection for performing sybil defense protocols fundamentally limits the ability of detecting sybil nodes. For example, nodes in a non-sybil community may mistake non-sybil nodes in another community for sybil nodes if the connections between two communities are sparse.

3. System Model

We model a P2P network as a link-weighted, directed graph $G=(V,E)$ with a finite set of vertexes and a set of links. A vertex represents a node, and an edge between two nodes reflects the history relation between two nodes in file sharing application. Each edge $e_{i,j}$is associated with a weight value, where $R_{i,j}$ is the number of service units that node I has purely contributed to node$j$. In other words, $R_{i,j}$is the number of service units that node i contributed to node j minus the number of units that node j contributed to node i. The value of$R_{i,j}$is stored by nodes on both sides, and $R_{i,j}=-R_{j,i}$.We assume that each node has a locally generated public/private key pair by which it can verify messages sent by neighbors. The detail of public key infrastructure is beyond the context of our paper. We assume that there are n common nodes, each of which maintain only an ID and obey the pre-established protocol. For clarity; we denote these common nodes contributing to the network at an acceptable level by sharing their resources with others as honest nodes. The network also consists of one or more malicious nodes which control a large number of IDs. We denote these malicious nodes and IDs as sybil nodes. Sybil nodes may consume a large number of service units from others and do not contribute to the network at an acceptable level. They may behave arbitrarily and collude with each other. As previous schemes [13-14], we denote the edge between an honest node and a sybil node as an attack edge. All nodes are determined either sybil nodes or non-sybil nodes. To simplify analysis, we assume that a node only can accept one service request in a time interval. If a node receives multiple requests, only the request with the highest priority is accepted. The goal of our protocols is two-fold. Our

protocols should restrict the number of service units consumed by sybil nodes to a reasonable level while reducing rejections of transactions among nodes for high efficiency. At a high-level, we try to isolate sybil nodes, separating sybil nodes from non-sybil nodes.

4. Sybil-Resilient Protocol

Our aim is not preventing adversaries from creating multiple identities, but that they are unable to obtain extra service units in doing so. The most ideal situation that we expect should satisfy the following two conditions(1): For the group of all non-sybil nodes, the total number of service units provided =the total number of service units obtained (2). For each non-sybil node, the node that provides more service units has a higher probability of obtaining service units than the node that provides fewer service units.

We explain why we do not think that the most ideal situation is that, for each of non-sybil nodes, the total number of service units provided = the total number of service units obtained. The reason for this is that some nodes that contribute a large amount of service do not require the corresponding amount of service. But a node which provides many service units should be assigned a high priority of obtaining service units.

To achieve the situation described above, we propose two contribution protocols: direct transaction and indirect transaction. Direction transaction only takes place between two adjacent nodes. Node *A* provides service to node *B* and then intends to obtain service from *B*. In this case, the node that provides service is the same as one that is interested in service later. The case of indirect transaction is more complicated in which the node that offers service is not the same as one that intends to obtain service. We define the two protocols below.

4.1. Direct Transaction

We distinguish two different modes in direct transaction: *supply* and *return*. Supply is the important mode by which a node may provide service to the other node. This is necessary to bootstrap the file sharing. Otherwise, no node can share files with anyone since initially no one has transaction records with others. It is worth emphasizing that the priority of supply is low. A node which does not receive requests of other modes, may accept the supply request. Let *T* denote the maximum weight value of each edge. When a node *A* receive a request of *t*service units from an adjacent node *B*, *A* provides service to *B* if $R_{A,B}+t \leq T$. After doing it, the value of weight $R_{A,B}$is added by *t*.

For example, there are two adjacent nodes *A* and *B* with $R_{A,B}$=0which means that both of them have no historical relations or the total number of service units that *A* provides to *B* = the total number of service units that *B* provides to *A*. As soon as *B* asks *A* to provide 5service units using supply mode, *A* may offer 5service units to *B*, and therefore $R_{A,B}$=5.

The second mode is return. If node *B* receives a request from node *A* that has offered *B* service units of *t* before ($R_{A,B}$=*t*), *B* should offer *s* service units to node *A* if *s* is no more than *t*. After doing it, $R_{A,B}$ will be updated. The priority of requests of return mode is high. The rationale of design behavior is obvious. The number of service units that a node offers to another node should be no more than the number of service units that the node received from it before.

Roughly speaking, here$R_{A,B}$= the number of service units that node *A* provided to *B* - the number of service units that *A* obtained from *B*. In order to obtain service using return mode, the node should have provided services before. For illustration, we use an example shown in Figure 1. Initially $R_{A,B}$=0. In Figure 1(a), *B* asks *A* to provide *5* service units. If A accepts the request, it provides *5* to *B*using the supply mode, and then$R_{A,B}$ =5. In Figure 1(b), *A* asks *B* to

provide *8* service units. Since *8* is more than$R_{A,B}$, *B* provides *5* service units using the return mode, and therefore$R_{A,B} = 0$. Whether or not to provide the rest of service units is decided by *B*. If *B* decides to use supply mode to accepts the request and$T \geq 3$, then $R_{A,B} = -3$.

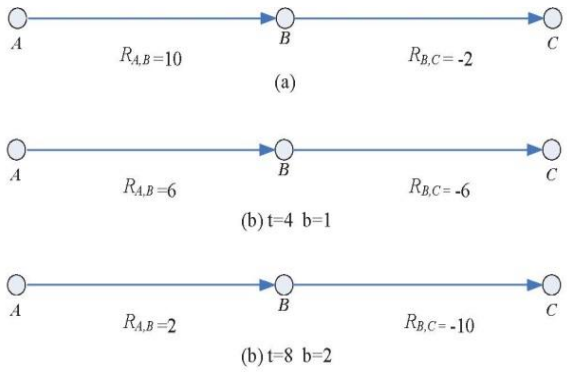


Figure 1. Direct Transaction Process

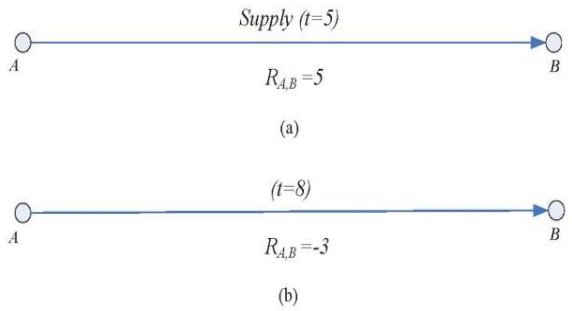


Figure 2. Indirect Transaction Process

4.2. Indirect Transaction

Unlike direct transaction, indirect transaction takes places in two non-adjacent nodes. There is only one mode: remote supply. When a node tries to obtain a certain number of service units from a remote node, it should determine whether or not there is a path between them in the directed graph. If there is, it may request service. Otherwise, and the request will be rejected by the remote node.

The process of indirect transaction consists of two phases: finding a path and updating weight values. Finding a transaction path includes running Dijkstra algorithm or other existing algorithms to route over the directed graph to determine a path P_{S-D}wherethe source *S*is a node that issues requests and the destination *D* is a node that provides service. Let R_{min}denote the lowest edge weight along the path.

After the transaction path P_{S-D} is available, the source node *S*easily knows the value ofR_{min}.We use *b*as a free parameter to fine-tune the behavior of the transaction of remote supply. Intuitively, the larger *b*, the higher ability of sybil attacks. On the other hand, a larger value of *b* also enhances the successful probability of obtaining service using remote supply mode.

If node *S* requests *t* service units from node *D*, it must guarantee that the condition $|R_{min}$-$t/b| \leq$ T holds for each edge on the path. Recall that *T* is the maximum value of weights. Each edge can be assigned an individually unique *T*. However, to simplify the protocol, we use the

same *T* for all edges in this paper. If *D* accepts the request and provides the service units, the weight value of each edge of the path is subtracted by*t*/*b*. The priority of the remote sypply mode is middle.

We illustrate the process by a network consisting of three nodes *A*, *B* and *C* as shown in Figure 2. Let *T*=10. First, *A* provides 10 to *B*, and *C* provides 2 to *B*, and thus $R_{A,B} = 10$ and $R_{B,C}$=-2 (Figure 2. (a)). If *A* wants to obtain service from *C*, it can use remote supply mode. According to our algorithm, R_{min}= -2. Without *b*, the maximum number of service units that *A* can request is 8. If *A* requests 4 and *C* accepts, then $R_{A,B}$=6 and $R_{B,C}$=-6 (Figure 2(b)). Next, we use *b* as a free parameter to fine tune the behavior of the remote supply. For *b*=2, the maximum number of service units that *A* can request is 8. If *A* requests 8 and *D* accepts, $R_{A,B}$=2 and $R_{B,C}$=-6-4=-10 (Figure 2(c)).

4.3. Priority

We now provide the rationale for defining the priority of each transaction mode. If node *A* gives service to *B*, *B* should return the same number of service units to *A* if it requires. Thus the priority of return is high. If node *A* give service to *B* and *A* give service to *C*, *C* owes *B* and *B* owes *A*, and consequently *C* should provide service to *A* to some extent. We explain why the priority of remote supply is middle, instead of high. If node *B* gives service to *C*, thus *C* trusts *B*. But *C* does not have a historical record about *A*. In some cases, the historical record, namely, the value of weight between *B* and *A* may be forged. Thus the priority of remote supply is lower than return mode. In supply mode, a node gives service to another node with which it has no historical records. Thus, the priority of supply mode is low. In summary, the priority of three modes from high to low is return, remote supply and supply.

4.4. Resistance to Sybil Attacks

We now provide the rationale for designing our protocols. Consider the network where a Sybil region *S* is attached to a non-sybil region *H*. There are several edges between the two regions, which are called attack edges. Since finding attack edges has been proved as an NP problem, attack edges are indistinguishable from normal edges between two non-sybil nodes. Instead of finding which one is an attack edge, we analyze the upper bound of the number of service units that the total sybil nodes can obtain according to our protocols.

Lemma 1: Let S_{SN} denote the number of service units that the sybil region *S* provides to the non-sybil region *N*, and S_{NS} denote the number of service units that non-syil region *N* provides to sybil region *S*. Our protocol can guarantee that $S_{NS} \leq b \times S_{SN} + a \times T$, where b is a free parameter, *a* is the number of attack edges and *T* is the maximum value of weights of all edges.

Proof: The weight of an attack edge reflects the number of service units that sybil nodes contribute to non-sybil nodes. Recall that the straightforward way of protecting from sybil attacks is that the weight of the attack edge is subtracted by *t* if sybil nodes obtain *t* service units from non-sybil nodes.

According to our protocols, if a sybil node obtains *t* service units from the non-sybil region, the weight of each edge in the transaction path will be subtracted by *t/b*. The transaction path undoubtedly includes the attack edge if it exists, thus the weight of the attack edges is reduced by *t/b*. After all weights of attack edges are reduced to zero, sybil nodes can at most obtain $a \times T$service units using either direct transaction or indirect transaction mode. Therefore the maximum number of service units that *S* can obtain from *H* is $b \times S_{SN} + a \times T$.■

In practice, the value of b is a constant value. In non-trivial cases, the value of S_{SN} is far more than the value of $a \times T$, thus S_{SN} is expected to dominate the equation. Note that we do

not make any assumptions regarding attack strategies that adversaries use and topological features. Therefore, the value of S_{NS} is mainly decided by the value of $b \times S_{SN}$ and independent of both attack strategies and topological features.

5. Dynamic Reputation Ranking Algorithm

To simplify the analysis, we assume that a node at most accepts one request at a time interval. During a time interval, each node may receive several demands from different nodes. Anode should accept the request with the highest priority. For example, a node receives two requests with different priorities, one is a return mode and the other is a remote supply mode. The node will accept the request of return mode and reject the request of remote supply mode.

There is still a question. For some cases, a node may receive multiple requests with the same priority from different nodes. How to choose suitable one among them? It is necessary to develop an algorithm to rate nodes, with which the node can distinguish the request of the node with the highest reputation. We present a dynamic reputation ranking algorithm, called *DRRA*, which holds sybil-resilient properties. A shortest-path algorithm is embedded in the algorithm. If node *S* intends to obtain service from node *D*, node *S* should find a shortest path to node *D* using the shortest-path algorithm. From the point of view of node *D*, the reputation rank value of node *S* is defined as rank $(P_{S-D})=min(\sum 1/(T+R_{i,j}) \mid e_{i,j} \in P_{S-D})$where P_{S-D} is the path from *S* to *D*. The reputation value corresponds negatively with the reputation level. This means that the higher the reputation value is, the lower the reputation level will be.

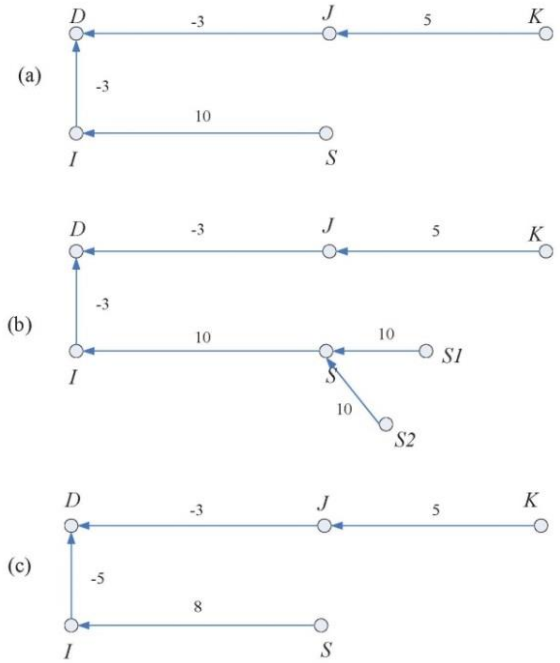


Figure 3. (a) Initial Status (b) Example of Sybil attacks (c) Status After Indirect Transaction

We use a simple example, illustrated in Figure 3 to show the algorithm. Let *T*=10 and *b*=1. The value above an edge is denoted as the weight value between two adjacent nodes. Suppose that both node *S*and node *K* try to obtain 2 service units from node *D* at the same time. Since

both of them have paths to node *D*, node *D* should calculate the reputation rank for each of them. We have

Rank (P_{K-D})=1/(10-3)+1/(10+5)=1/7+1/15

Rank (P_{S-D})=1/(10-3)+1/(10+10)=1/7+1/20

Obviously, rank (P_{K-D})>rank(P_{S-D}). As a result, node *S* has priority to obtain service from node *D* instead of node *K* (Figure 3 (a)). We observe that the node providing more resources to others will maintain more paths to others, and therefore it has more probability of obtaining service. For this reason, we can see that our algorithm operates as an incentive mechanism of encouraging nodes to contribute more resources.

In what follows, we show that the ability of sybil attacks is restricted by the property of sybil-resilience of our algorithm. Suppose that sybil nodes *S1*, *S2* arbitrarily modify weight values among them. Both rank (P_{S1-D}) and rank (P_{S2-D}) are always larger than rank (P_{S-D}), thus *S* obtains service instead of *S1* and *S2* (Figure 3 (b)). According to the indirect transaction protocol, the contribution value is subtracted by 2 from each hop in the transaction path if *b*=1. The statuses after completing a transaction are shown in Figure 3 (c). In order to theoretically analyze the property of our algorithm, we take advantage of the research result [12] in which they prove that a protocol is sybil-resilient if the protocol holds the properties of diminishing returns, monotonicity and no splitting. We can easily prove that DDRA have such properties. Due to limited size of the paper, we omit the proving process.

In existing schemes, reputations are global, in the sense that every node has the same view of a given node's reputation. However, in our algorithm, reputations are function of both the source node and the destination node. A node that has a low reputation for a destination node may have a high reputation for another. Marginally lower reputed nodes which are honest will have a chance to take part in a transaction, improving their reputations. Thus, there are few cases of starving for those nodes.

6. Evaluation

In this Section, we use a simulation-based approach to evaluate our protocol from various aspects and in different scenarios. First we describe the simulation mode and metrics. We implement our protocols on Gnutella simulation tool which is an event-driven simulator using the C++ programming language on windows 7 platform. Routing protocols and interactions among nodes are based on the Gnutella protocol specification given in [19].

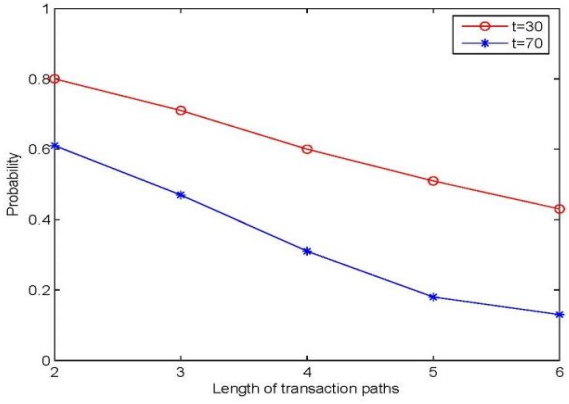


Figure 4. Pdf of Successful Transactions for *b*=1

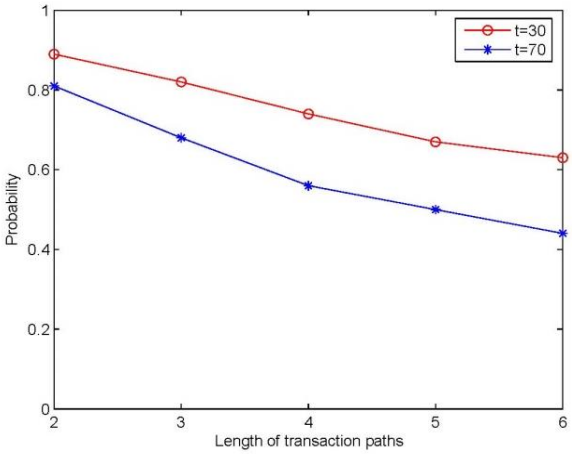


Figure 5. Pdf of Successful Transactions for *b*=2

We generate two random synthetic networks, each of which consists of 2^{12}nodes without any community structures.The initial value of degree *d*=8. One is defined as a non-sybil region, and the other is defined as a sybil region. To evaluate the performance of our sybil defense protocol, we randomly add several edges to connect two regions. As a result, the sybil region is attached to the non-sybil region by these edges. A simulation experiment is repeated 100 times and the result is an average of the results of 100 individual runs.

6.1. Probability of Successful Transactions

We evaluate the efficiency of file sharing. In each simulation run, we randomly generate a transaction path in non-sybil region, where the initial weight value of each edge is chosen uniformly at random from -100 to 100.The maximum weight value of each edge, *i.e.*, *T* is 100. Each source node requests *t* service units, where *t* is varied from 30 to 70.

If the weight values of all edges in a path satisfy the requirement of remote supply transaction, the transaction process will be regarded as a successful case. Otherwise, it will be regarded as a failed case. The probability of successful cases is an important metric indicating how many nodes can obtain service successfully. We measure the maximum number of service units received as a function of the length of transaction path (*l*). Figure 4 and 5 show the probability distribution of successful transactions for *b*=1and 2 respectively.

We observe that the performance of file sharing is sensitive to the length of transaction path.As *l* increases, the probability of successful transactions falls. To achieve a good performance, we should increase *b*.When *b*=2, more than 40 percent of service requests satisfy the condition of remote supply even if *l*=6. Simulation results show that our protocol is efficient in terms of file sharing.

6.2. C**apacity of Sybil Attacks**

We are interested in exploring *R*, the proportion of the total number of service units that sybil nodes (S_{NS}) receive vs. the total number of service units that sybil nodes contribute to (S_{SN}).The *R* ratio represents the capacity of sybil attacks. A larger *R* means a higher capacity of sybil attacks.

In this simulation, we assume that sybil nodes totally contribute 5000 service units to non-sybil nodes for creating attack edges, thusS_{SN}=5000. Note that it is necessary that sybil nodes

provide service to non-sybil nodes, otherwise there are no edges between sybil nodes and non-sybil nodes.

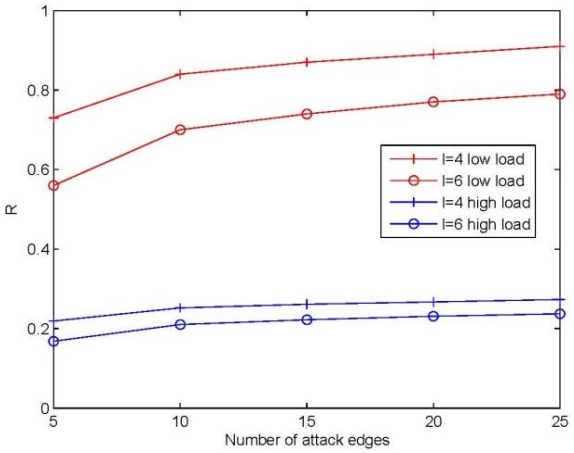


Figure 6. Capacity of Sybil Attacks (*b*=1)

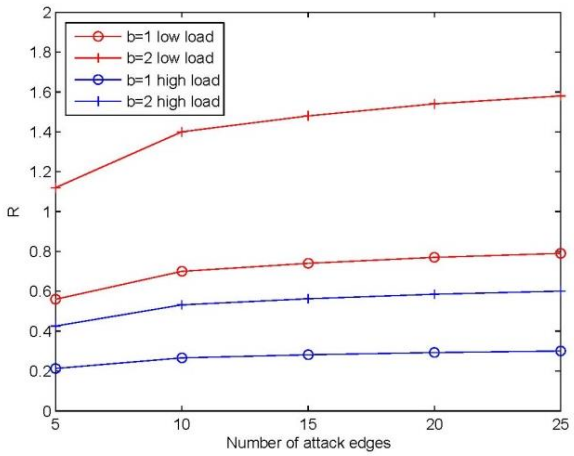


Figure 7. Capacity of Sybil Attacks (*l*=6)

To evaluate our protocols, we consider two cases: low load and high load.In the case of low load, only one syibl node asks a remote non-sybil node to provide service. Thus we simply use indirect transaction protocol without DRRA.In the case of high load; a non-sybil node receives multiple requests from several nodes in the same interval. We assume that one non-sybil node and one sybil node ask a non-sybil node to provide service at the same time. Both of the two consumer nodes are chosen randomly from the non-sybil region and sybil region respectively. Thus we combine indirect transaction protocol with DRRA to calculate the reputation value for each node.

We measure *R* with different number of attack edges which are varied from 5 to 25, and *b*=1. Figure 6 shows that the capacity of sybil attacks is positive correlated with the number of attack edges. We observe the capacity of sybil attacks grows slowly if the number of attacks edges is more than 15. We measure *R* for varying values of *b*. The length of transaction paths (*l*) is 6. As the number of attack edges increases, the value *R* grows slowly. As observed from Figure 7, in both low load and high load, the value of *R* is less than *b*. This means that the

total number of service units that sybil nodes obtain is the order of the total number of service units that sybil nodes contribute to.

Figure 6 and Figure 7 illustrate that the value of *R* of high load is significantly less than that of low load when the value of *b* is fixed.Sybil nodes cannot obtain service due to failing to complete with non-sybil node. This shows DRRA is indeed necessary. The simulation results also correspond well with lemma 1, and therefore our protocols defeat sybil attacks successfully.

7. Conclusions

This paper presents a sybil-resilience transaction protocol and a dynamic reputation rating algorithm. Combining these two, we can achieve a desirable level of file-sharing efficiency while defeating sybil attacks. Two key factors in analyzing sybil defense protocols should be taken into consideration. One is fairness; we expect that sybil nodes cannot obtain service units from non-sybil nodes. The other is efficiency; a non-sybil node can obtain service from other nodes without any pre-conditions. The further analysis of the trade-off between the capability of sybil attacks and the capability of transactions maybe a good future work.

Acknowledgements

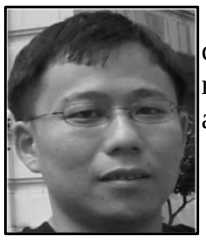
This work was supported by the National NSF of China (NO. 61272315,NO.60842009) and the Zhejiang provincial Natural Science Foundation of China (No. LY12H29012).

References

[1] J. R. Douceur, "The Sybil attack", In the first International Workshop on Peer-to-Peer Systems, **(2002)**.
[2] B. Cohen, "Incentives build robustness in bittorrent", In Proceedings of P2PEcon'07, **(2003)**.
[3] S. D. Kamvar, M. T. Schlosser and H. Garcia-Molina, "The Eigen Trust algorithm for reputation management in P2P networks, In Proceedings of WWW'03, **(2003),** pp. 640-651.
[4] R. Gupta and A. K. Somani, "Reputation management framework and its use as corrency in large-scale peer-to-peer network's", In Proceedings of IPTPS, **(2002)** March.
[5] J. R. Douceur, "Reputation management framework and its use as currency in large-scale peer-to-peer network's", In Proceedings of P2P, **(2004),** pp. 124-132.
[6] B. Viswanath, A. Post, K. P. Gummadi and A. Mislove, "An analysis of social network-based sybil defenses", In Proceedings of SIGCOMM, **(2010)**.
[7] K. Hildrum and J. Kubiatowicz, "Asympotically efficient approaches to fault-tolerance in peer-to-peer network's", In Proceedings of 17th International Symp. On Distributed Computing (DISC), **(2003),** pp. 321-336.
[8] M. Castro, P. Druschel, A. Ganesh, A. Rowstron and D. S. Wallach, "Secure routing for structured peer-to-peer overlay networks", In Proceeding of the 5th Symp. on Operating Systems Design and Implementation(OSDI), **(2002),** pp. 299-314.
[9] H. Rowaihy, W. Enck, P. McDaniel and T. L. Porta, "Limiting Sybil attacks in structured P2P networks", In Proceedings of INFOCOM, **(2007),** pp. 2596-2600.
[10] N. Borisov, "Computational puzzles as Sybil defenses", In Proceedings of the 6th IEEE Int'l Conf. on Peer-to-Peer Computing, **(2006),** pp. 171-176,
[11] I. Baumgart and S. Mies, "S/Kademlia: A praticable approach towards secure key-based routing", In Proceedinds of the 13th International conference on Parallel and Distributed Systems, **(2007),** pp. 1-8.
[12] A. Cheng and E. Friedman, "Sybilpro of reputation mechanisms", In Proceedings of P2PEcon'05, **(2005),** April, pp. 128-132.
[13] H. F. Yu, M. Kaninsky and P. B. Gibbons, "Sybil limit: A near-optimal social network defense against sybil attacks", In Proceeding of SIGCOMM, **(2006)**.
[14] H. F. Yu, M. Kaninsky and P. B. Gibbons, "Sybil limit: A near-optimal social network defense against sybilattacks", In Proceedings of IEEE Symposium on Security and Privacy, **(2008)**.
[15] W. Wei, F. Y. Xu, C. C. Tan and Q. Li, "Sybil Defender: Defend Against Sybil Attacks in Large Social Networks", In INFOCOM, **(2012)**.

[16] G. Danezis and P. Mittal, "Sybil Infer: Detecting sybil nodes using social networks", In Proceedings of NDSS, **(2009)**.
[17] N. Tran, B. Min, J. Li and L. Subramanian, "Sybil-Resilient Online Content Voting", In Proceedings of the 6th USENIX Symposium on Networked Systems Design and Implementation (NSDI), **(2009)** April.
[18] N. Tran, J. Li, L. Subramanian and S. S. M. Chow, "Optimal Sybil-resilient Node Admission Control", In Proceedings of the 30th IEEE International Conference on Computer Communications (INFOCOM), Shanghai, China, **(2011)** April.
[19] "Clip2, The Gnutella Protocol Specification v0.4(Document revision 1.2)", **(2001),** http://www9.limewire.com/developer/gnutellaprotocol0.4.pdf.

Author



Xu Xiang, he is a member of the ACM, and an academic staff in the college of information engineering of China Jiliang University. His research interests included is tribute and parallel computing systems and computer networks, with an emphasis on peer-to-peer network.

