

Anti-Sybil Mechanism against Bogus Identities in Social Networks

G. Lawrence Paul Sundararaj¹ D. R. Anita Sofia Liz²

PG Scholar, Department of IT, Dr.Sivanthi Aditanar College of Engineering, Tiruchendur, India¹

Assistant Professor, Department of IT, Dr.Sivanthi Aditanar College of Engineering, Tiruchendur, India²

Abstract: Most of the large scale social networking sites and small private social networks on the Internet are open to Sybil attacks. The Sybil attack is an attack where in an adversary creates multiple Duplicate or False identities to compromise the running of the system. By including false information by the Duplicated entities, an adversary can mislead a system into making decisions benefiting. Defending against Sybil attacks is quite challenging. This paper presents Sybil Defender, a sybil defense mechanism that leverages the network topologies to defend against sybil attacks in social networks. Sybil Defender can effectively identify the sybil nodes and detect the sybil community around a sybil node and it is feasible to limit the number of attack edges in online social networks by relationship rating.

Keywords: Sybil attack, social network, random walk

I. INTRODUCTION

Most networks, like a peer-to-peer network, rely on assumptions of identity, where each computer represents one identity. A Sybil attack happens when an insecure computer is hijacked to claim multiple identities, in other words, a Sybil attack [1] occurs when the attacker creates multiple identities (sybils) and exploits them in order to manipulate a reputation score. Problems arise [2] when a reputation system (such as a file-sharing reputation on a torrent network) is tricked into thinking that an attacking computer has a disproportionately large influence. Similarly, an attacker with many identities can use them to act maliciously, by either stealing information or disrupting communication. It is important to recognize a Sybil attack and note its danger in order to protect yourself from being a target.

Recently, there has been an increasing interest in defending against sybil attacks in social networks [3], [4], [5], [6], [7]. In a social network, two user identities share a link if a relationship is established between them. Each identity is represented as a node in the social graph. To prevent the adversary from creating many sybil identities, all the previous sybil defense schemes are built upon the assumption that the number of links between the sybil nodes and the honest nodes, also known as attack edges, is limited. As a result, although an adversary can create many sybil nodes and link them in an arbitrary way, there will be a small cut between the honest region and the sybil region. The small cut consists of all the attack edges and its removal disconnects the sybil nodes from the rest of the graph, which is leveraged by previous schemes to identify the sybil nodes. Note that the solution to this problem is nontrivial, because finding small cuts in a graph is an NP-hard problem. To limit the number of attack edges, previous schemes assume that all the relationships in social networks are trusted and they reflect the trust relationships among those users in the real world, and thus, an adversary

cannot establish many relationships with the honest users. However, it has been shown that this assumption does not hold in some real-world social networks [8].

To address the problem Sybil Defender is introduced, which is a centralized sybil defense mechanism. It consists of a sybil identification algorithm to identify sybil nodes, a sybil community detection algorithm to detect the sybil community surrounding a sybil node, and two approaches to limiting the number of attack edges in online social networks. Our scheme is based on the observation that a sybil node must go through a small cut in the social graph to reach the honest region. An honest node, on the contrary, is not restricted. Now, if we start from a sybil node to do random walks, the random walks tend to stay within the sybil region. The main contributions of this work include:

- i. Based on performing a limited number of random walks within the social graphs, our proposed sybil identification and sybil community detection algorithms are more efficient than previous techniques for large social networks.
- ii. We evaluate SybilDefender using two large-scale social network samples from Orkut and Facebook, respectively. The results show that the performance of our sybil identification algorithm approaches the theoretical bound, and it outperforms SybilLimit, the state-of-the-art sybil defense mechanism that applies to large social networks, by more than 10 times in both accuracy and running time. In addition, our sybil community detection algorithm can effectively detect the sybil community around a sybil node with short running time.

We propose two practical techniques to limit the number of attack edges in online social networks, and develop a Facebook application to demonstrate the feasibility of one of

the techniques. The survey results of our Facebook application show that the assumption made by previous work that all the relationships in social networks are trusted does not hold in online social networks, and it is feasible to limit the number of attack edges in online social networks by relationship rating.

II. RELATED WORK

Sybil Guard [7] and Sybil Limit [6] are among the first Sybil detection schemes to be proposed. Sybil Guard uses the intersections between modified random walks to determine whether identities should be given access to the system. Sybil Limit improves on Sybil Guard's bound by using multiple walks, which allows it to accept fewer Sybil identities per attack edge. Both of these schemes can be implemented in a centralized or decentralized fashion.

Sybil Infer [3] is a centralized protocol that assumes full knowledge of the social graph. It uses a Bayesian inference technique that assigns to each node its probability of being Sybil. Unlike Sybil Guard and Sybil Limit, Sybil Infer does not provide any theoretical bounds on the number of Sybil identities accepted per attack edge. In the evaluation, Sybil Infer handled networks with up to 30,000 nodes, which is much smaller than the size of regular online social networks.

Gate Keeper [23] is a decentralized Sybil detection protocol that improves over the guarantees provided by Sybil Limit. It uses a variant of the ticket distribution algorithm used in Sum Up [22] from multiple random identities in the graph to detect Sybils. Even though social network-based Sybil detection schemes are relatively simple and easy to integrate into the system, they all suffer from inherent limitations. In particular, these schemes make strong assumptions about the topology of the social graph, where many of real-world social networks do not conform to these assumptions. Consequently, these schemes have not found mainstream adaption, and they usually result in high false positive and false negative rates in real world social networks. In contrast, Sybil Defender only relies on performing a limited number of random walks in the social graph, and it is scalable to large networks.

III. PROPOSED FRAMEWORK

We denote the social network as a graph G consisting of vertices V and edges E . There are n honest users in the social network, each with one identity, denoted as an honest node in V . There are also one or more malicious users in the social network, each with a number of Sybil identities. Each Sybil identity is denoted as a Sybil node in V . A relationship between two identities in the social network is represented as an edge connecting the two corresponding nodes in G . The edges in G are undirected. We name the edge between a Sybil node and an honest node an attack edge. The Sybil region consists of all the Sybil nodes, while the honest region consists of all the honest nodes. All the Sybil nodes are controlled by an adversary. Thus, the adversary can create

arbitrary edges within the Sybil region. Graphs [10] are used to represent relationships or connections (edges) between domain objects (vertices). Social networks are an application of a graph data structure.

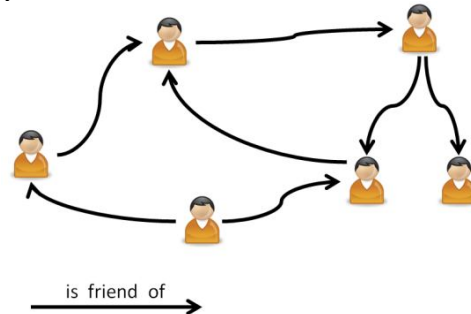


FIGURE 1 A Social Network

The proposed methodology, Sybil Defender is a famous approach for anti-sybil. It consists of two steps: Sybil node detection and Sybil Community detection (Fig 2). One way to defend against sybil attacks in social networks is to leverage the social network topologies. This paper is build on the following assumptions:

The honest region is fast mixing, which means a random walk of length $O(\log n)$ is long enough such that with probability at least $1-1/n$, the last traversed node is drawn from the node stationary distribution of the graph.

1. One honest node is known.
2. The social network topology is known
3. The size of the sybil region is not comparable to the size of the honest region.
4. the number of attack edges is limited.

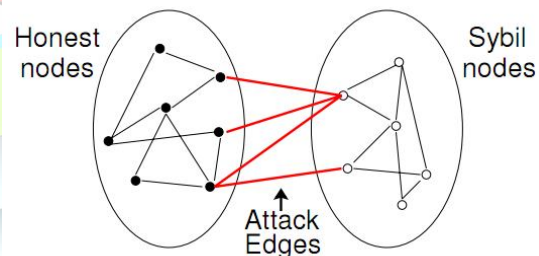


FIGURE 2 an honest community and Sybil community

In a social network, the vertices (nodes) are identities in the distributed system and the (undirected) edges correspond to human-established trust relations in the real world. The edges connecting the honest region (i.e., the region containing all the honest nodes) and the Sybil region (i.e., the region containing all the Sybil identities created by malicious users) are called attack edges.

A. Sybil node detection

In Sybil node detection, the Sybil identification algorithm that takes the social graph $G(V, E)$, a known honest

node h , and a suspect node u as input, and outputs whether u is Sybil or not. Our algorithm is based on random walks. A random walk on a graph is defined by the sequence of moves of a particle between nodes of G .

For a suspect node, based on preknown honest nodes' statistical features, Sybil Defender determines whether the suspect node is a Sybil or not. After finding a Sybil node, based on the assumption that Sybil nodes are more likely to connect with other Sybil nodes, the defence will detect the Sybil community in which the Sybil node resided.

This defense is based on two assumptions:

(1) the number of links between honest users and Sybils is limited and

(2) the size of the Sybil community is smaller than that of millions of users: for the attacker, to register such a large number of identities is impossible.

From an honest user, we can send a fixed number of random walkers to pass an l -length random path, assuming there are k walkers. At other nodes, we can compute the times that these random walkers passed through this node and call the times their *visiting frequency*. After that, we can calculate the statistical distribution of the visiting frequency. If the random walks from a suspect node do not follow some statistic distribution, then the suspect is Sybil.

B. Sybil community detection

After one Sybil node is identified, our Sybil community detection algorithm can be used to detect the Sybil community surrounding it. The Sybil community detection algorithm takes the social graph $G(V, E)$ and a known Sybil node s as input, and outputs the Sybil community around s . Our algorithm relies on performing partial random walks originating from s . Each partial random walk behaves the same as the simple random walks used in the previous section, except that it does not traverse the same node more than once. Therefore, when a partial random walk reaches a node with all the neighbors traversed by it, this partial random walk is "dead" and cannot proceed. Figure 3 illustrates the Sybil Defender. Suppose that we have already known an honest node. From this node, we send out k random walks with a fixed length l . Since social network (in the honest region) is fast mixing, which means that any pair of nodes can reach one another at an $O(\log n)$ -length random path, a circle region in the honest community will be covered by the random walk. However, because the size of the Sybil community is smaller than that of honest one, the majority of random walks in the Sybil region is different from the honest one. By this way, a suspect node can be verified.

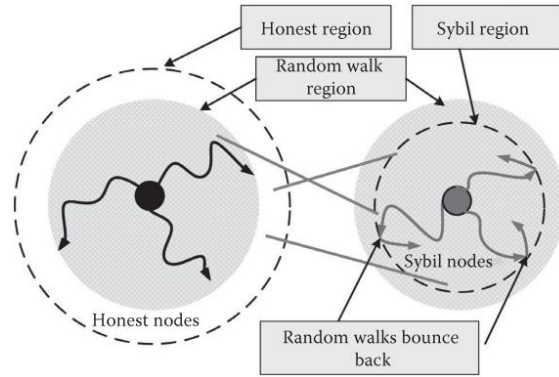


FIGURE 3 The idea of Sybil Defender.

After finding a Sybil node, the Sybil Defender can also detect its resident Sybil community, based on the fact that Sybil nodes are more likely to connect with other Sybil nodes. The detection of a Sybil community can be done by using loop-free random walks. Consider that when a random walker passes the same node twice, it means the random walkers reach the boundary of the Sybil community. Sybil Defender renders a random walker dead if it arrives at the same node twice. Similar to the process of verifying a suspect node, Sybil Defender also initializes several random walks with different lengths. Again, the reason is that the size of the Sybil community is unavailable. If the dead ratio of the L -length group of random walks is greater than a predefined threshold, then all the passed nodes will be regarded as members of a Sybil community.

Algorithm 1 is used to calculate the mean number of nodes with frequency no smaller than t when performing R random walks in length of l originating from known honest node h . The larger the community is, the larger mean number it gets.

Algorithm 1. PreProcessing(G, h).

```

1:  $J = \{h\}$ 
2: for  $i = 1$  to  $f$  do
3:   Perform a random walk with length  $l_s = \log n$ 
   originating from  $h$ 
4:    $J = J \cup \{\text{the ending node of the random walk}\}$ 
5: end for
6:  $l = l_{min}$ 
7: while  $l \leq l_{max}$  do
8:   for  $i = J.first()$  to  $J.last()$  do
9:     Perform  $R$  random walks with length  $l$ 
     originating from node  $i$ 
10:    Get  $n_i$  as the number of nodes with frequency no
    smaller than  $t$ 
11:   end for
12:   output  $\langle l, \text{mean}(\{n_i : i \in J\}), \text{stdDeviation}(\{n_i : i \in J\}) \rangle$ 
13:    $l = l + 100$ 
14: end while

```


Algorithm 2 is to detect whether a given node is a sybil node or not. As is said in above, if a node has a very low number, it may be a sybil node.

Algorithm 2. SybilIdentification($G, u, \text{tuples from Alg.1}$).

```

1:  $l = l_0$ 
2: while  $l \leq l_{max}$  do
3:   Perform  $R$  random walks with length  $l$  originating from  $u$ 
4:    $m =$  the number of nodes whose frequency is no smaller than  $t$ 
5:   Let the tuple corresponding to length  $l$  in the outputs of Algorithm 1 be  $\langle l, mean, stdDeviation \rangle$ 
6:   if  $mean - m > stdDeviation * \alpha$  then
7:     output  $u$  is sybil
8:   end the algorithm
9: end if
10:   $l = l * 2$ 
11: end while
12: output  $u$  is honest

```

Partial random walk is a little different from regular random walk: once a node is 'walked', it can not be walked once again. So a partial random walk can be terminated when a node have no neighbors to walk into without reaching a given length. It is called a 'dead walk'. Given a sybil node s , we can estimation a length when the dead Walk Ratio is smaller than a given threshold. All the nodes within a length less than the estimated length are suspected to be sybil nodes.

Algorithm 3. WalkLengthEstimation(G, s).

```

1:  $l = l_0/2$ 
2:  $deadWalkRatio = 0$ 
3: while  $deadWalkRatio < \beta$  do
4:    $l = l * 2$ 
5:    $deadWalkNum = 0$ 
6:   for  $i = 1$  to  $R$  do
7:     Perform a partial random walk originating from  $s$  with length  $l$ 
8:     if the partial random walk is dead before it reaches  $l$  hops then
9:        $deadWalkNum++$ 
10:    end if
11:  end for
12:   $deadWalkRatio = deadWalkNum/R$ 
13: end while
14: output  $l$ 

```

Rate all the suspected node based on conductance, defined as follows. Let d be the sum of the degrees of all the nodes in set S , and a be the number of edges with one endpoint in S and one endpoint in S' . The the conductance of S is a/d .

Based on the assumptions, there should be a small cut between the honest region and sybil region. The conductance of the sybil region is very small. So it is reasonable to leverage a greedy algorithm to detect which nodes are real sybil nodes.

Algorithm 4. SybilRegionDetection(G, s, l from Alg.3).

```

1: Set the frequency of all the nodes to be 0
2: for  $i = 1$  to  $R$  do
3:   Perform a partial random walk originating from node  $s$  with length  $l$ 
4:    $s.frequency++$ 
5:   for  $j = 1$  to  $l$  do
6:     Let the  $j^{th}$  hop of the partial random walk be node  $k$ 
7:      $k.frequency++$ 
8:   end for
9: end for
10:  $traversedList =$  Sort the traversed nodes by their frequency in decreasing order
11:  $counter = 0$ 
12:  $S = \emptyset$ 
13: do
14:    $counter = conductance(S)$ 
15:   for  $i = traversedList.first()$  to  $traversedList.last()$  do
16:     if node  $i \in S$  then
17:       continue
18:     if  $conductance(\{i\} \cup S) \leq conductance(S)$  then
19:        $S = \{i\} \cup S$ 
20:   while ( $counter > conductance(S)$ )
21: output  $S$ 

```

The Sybil nodes tend to be in front of the honest nodes in the sorted list, because a large number of partial random walks cannot enter the honest region, due to the existence of the small cut between the honest region and the Sybil region. This algorithm only relies on performing R partial random walks originating from a Sybil node, which makes it very efficient and scalable to large-sized social networks.

IV. EXPERIMENTAL RESULTS

We study the behavior of Sybil Defender when there are malicious users. In most security research, the term "malicious user" typically refers to a single malicious user who does not assume additional identities. In random, we repeatedly pick uniformly random nodes in the graph as Sybil attackers, until the total number of attack edges reaches a certain value. The experimental results of our proposed methodology show the accuracy and the efficiency level a bit higher than the existing ones. Also the Sybil node detection rate and time consumption also tends to be high. False identification rate are very less of probability with our Sybil Defender. We have obtained all corresponding results for Sybil community as well, which are always slightly better but the difference is usually negligible.

V. CONCLUSION

The challenge with the Sybil Defender is that how to extract the correct visiting frequency distribution from the honest region. This paper presented Sybil Defender, a centralized Sybil defense mechanism against sybil attacks

using social networks. Sybil Defender consists of a sybil identification algorithm, a sybil community detection algorithm, and two approaches to limiting the number of attack edges in online social networks. Our results on social networks confirmed their fast mixing property, and thus validated the fundamental assumption behind this approach. As future work, we intend to implement Sybil Limit within the context of some real-world applications and demonstrate its utility.

ACKNOWLEDGEMENT

I extend my sincere thanks to D. R. Anita Sofia Liz for her innovative ideas and suggestions throughout my project and the successful completion of my project.

REFERENCES

- [1] http://itlaw.wikia.com/wiki/Sybil_attack.
- [2] <http://anti-virus-software-review.toptenreviews.com/what-is-a-sybil-attack-.html>.
- [3] G. Danezis and P. Mit, "Sybilinfer: Detecting Sybil Nodes Using Social Networks," Proc. Network and Distributed System Security Symp. (NDSS), 2009.
- [4] N. Tran, J. Li, L. Subramanian, and S.S. Chow, "Optimal Sybil-Resilient Node Admission Control," Proc. IEEE INFOCOM, 2011.
- [5] L. Xu, S. Chainan, H. Takizawa, and H. Kobayashi, "Resisting Sybil Attack by Social Network and Network Clustering," Proc. IEEE/IPSJ 10th Int'l Symp. Applications and Internet (SAINT), 2010.
- [6] H. Yu, P.B. Gibbons, M. Kaminsky, and F. Xiao, "SybilLimit: A Near-Optimal Social Network Defense against Sybil Attacks," Proc. IEEE Symp. Security and Privacy, 2008.
- [7] H. Yu, M. Kaminsky, P.B. Gibbons, and A. Flaxman, "SybilGuard: Defending against Sybil Attacks via Social Networks," Proc. ACM SIGCOMM, 2006.
- [8] L. Bilge, T. Strufe, D. Balzarotti, and E. Kirda, "All Your Contacts Are Belong to Us: Automated Identity Theft Attacks on Social Networks," Proc. 18th Int'l Conf. World Wide Web (WWW '09), 2009.
- [9] 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'09, pages 15–28, Berkeley, CA, USA, 2009. USENIX Association.
- [10] <https://blog.codecentric.de/en/2012/02/spring-data-neo4j/>

About Author



G. Lawrence Paul Sundararaj received his B.Tech degree in Information Technology from Dr. Sivanthi Aditanar College of Engineering, Tiruchendur in 2012. Currently, he is pursuing M.Tech degree in Information Technology in Dr. Sivanthi Aditanar College of Engineering, Tiruchendur. His research areas of interest include Network Security, Data Mining and Knowledge Engineering.

D. R. Anita Sofia Liz received his B.E degree in Computer Science from PSR Engineering College in 2004 and received her M.E degree in Computer Science from Dr. Sivanthi Aditanar College of Engineering, Tiruchendur in 2014. She has 10 years and 1 month experience as Assistant Professor and her research areas are Network Security, Image Processing and Operating Systems.