This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE ICC 2011 proceedings

Sybil Detection via Distributed Sparse Cut Monitoring

Aditya Kurve and George Kesidis (*Senior Member, IEEE*)
EE and CSE Departments, The Pennsylvania State University, University Park, PA, 16802
{ack205, gik2}@psu.edu

***Abstract*—Decentralized reputation systems help to enforce discipline and fairness in large unstructured and ad-hoc systems by rewarding good behavior and penalizing dishonest or greedy behavior. They are essential in large networks of independent nodes where centralized monitoring of node behavior is difficult due to the sheer size of the network. Sybil nodes pose a threat to the reputation systems by false referrals through sybil identities. We propose a scalable and distributed algorithm to identify attack edges and quarantine sybil clusters. This algorithm works well with dynamic trust graphs as nodes do not need to store any pre-computed data.**

***Index Terms*—sybils, reputation, peer to peer, social networks**

I. Introduction

A reputation system can track the history of node behavior in a P2P system of interacting agents or nodes. A reputation between two peers represents some degree of trust between them. "Direct" reputation is based on the somehow accumulated transaction outcomes between two peers. Reputation is relative to the observing node, since a transaction outcome might differ in the level of satisfaction between the parties involved. As two peers interact, each of them accrues some level of reputation from the point of view of the other. The relation between the reputation value and a measure of transaction success is subject to interpretation. Some have proposed using the aggregate or average reputation of a node over other nodes. In the case of an absence of any direct trust relationship between two peers, the aggregate reputation of a node can be used as a measure to judge [11]. However, such a mechanism involves the challenge of aggregating the distributed reputation values via some kind of consensus propagation. Instead of calculating aggregate reputation values, referral schemes can be used. In this case, if direct reputation is absent then nodes chain reputations in referral paths, *e.g.,* [15]. Referrals work in parallel with the (direct) reputation system to allow new peers of the group to more quickly ascertain trust relationships with other peers.

Reputation networks are prone to sybil attacks [10] wherein an individual acquires a large group of fake identities with a malicious or selfish intent. The sybil attacker can defame honest peers, or it can improve its own reputation using these fake sybil nodes, *e.g.*, by recommending its sybil nodes to honest nodes as extremely trustworthy peers. Alternatively, a sybil attacker might free ride by accumulating small initial resources given to putative newly arrived peers to a group (initialization attack).

Proposed approaches to sybil defense employ several different techniques. Some methods propose to verify every node's credentials by leveraging a one-to-one correspondence to its IP address [7] or its physical location [3]. The cost of identity acquisition can be increased by, *e.g.*, demanding the solution of a computational puzzle beforehand or a visual challenge-response [20] (the latter ordinarily used to discriminate a human correspondent from an automated bot). Defenses based on social networks have been considered in [21][8][19]. Decentralized methods are desired for unstructured, ad-hoc networks, but guaranteed protection is difficult in decentralized schemes. In this paper we leverage the idea of [21] and propose a scalable and distributed algorithm that exploits it, where "social trust" is interpreted here as that obtained through on-line transactions and associated referrals.

We assume that the graph of direct reputation, linking a cluster of honest nodes, has dense connectivity* or a high value of "sparse cut" (defined below). This implies that a substantially large cluster of connected nodes can be separated from the rest of the graph by a proportionally large number of edges. This property fails when sybil nodes are present because a direct reputation link with an honest node is difficult to establish compared to a referral link. Hence a sybil cluster is typically sparsely connected via direct reputation links with the rest of the reputation graph [21]. And so identifying a sparse cut in a reputation graph will help us to identify sybil nodes.

In Section II, we describe our model of the social network and reputation system and introduce a weighted sparsity of the graph cut as grounds for suspicion. We also introduce "police nodes" in this section. The algorithm relies on these nodes to obtain untampered data on reputation and trust relationships for the algorithm to work on. Section III describes the background of the proposed algorithm and examines the hierarchical paradigm. Section IV describes the algorithm in detail. Simulation results are given in Section V. We conclude in Section VI.

II. Network Model and Sparse Cuts

A. Sparse cuts

Consider a connected graph $G = (V, E)$ where V is the set

This work is supported by the NSF under grant number 0916179.

*[21] refers to this property as fast mixing property

of peers and E is the set of edges representing direct reputation links between them. For the moment we will ignore the relationships due to referral paths. The nodes and the links in the graph are weighted, and the weight of a link is denoted $w(u,v)$, where $u,v \in V$. Due to the asymmetric nature of trust relationships, $w(u,v) \neq w(v,u)$ generally. The weight of a cluster S, denoted $|S|$, is the combined weight of all the nodes in the cluster. Suppose we cut (partition) the graph G into two sub-graphs with A and A' representing the two sets of connected nodes in the two partitions. The *sparsity* of this cut is given by:

$$\tilde{C}(A) = \frac{\sum_{(u,v):\ u\in A, v\in A'} w(u,v)}{\min(|A|,|A'|)} \qquad (1)$$

where $w(u,v)=0$ if $(u,v) \notin E$. This definition is similar to the Cheeger Isoperimetric Constant in Riemannian geometry [4]. We can slightly modify the above definition of sparsity of a cut in order to accommodate the asymmetric nature of trust relationships:

$$C(A) = \min\left(\frac{\sum_{(u,v):\ u\in A, v\in A'} w(u,v)}{|A'|}, \frac{\sum_{(v,u):\ v\in A', u\in A} w(v,u)}{|A|} \right) \qquad (2)$$

Note that $C(A)=C(A')$. The sparsity of the graph G is the minimum value of sparsity considering all the different ways in which we can partition the graph:

$$S(G) = \min_{A\subset V} C(A) \qquad (3)$$

Our assumption that reputation graphs consisting of only honest nodes will be densely connected is based on [9], which states that dense connectivity can be a consequence of the presence of long-range links in graphs. In on-line peer-to-peer (P2P) social networks, if peer nodes have a propensity to establish new trust links with nodes other than those close to their own neighbors, the trust graph will be densely connected. Also, dense connectivity can be a consequence of the existence of a sufficiently dense set of highly connected super-peers or hubs.

B. Identifying Sparse Cuts

The problem of finding sparse cut in a graph is closely related to that of finding a minimum cut in a graph. The theory of min-cuts has been extensively studied for the purposes of simulation of large networks on parallel machines, routing in VLSI circuits, and parallel computation including sparse matrix analysis. This problem is known to be NP-complete, so several heuristic approaches have been developed, *e.g.* [12], [16], [13] including search by Markov chain Monte Carlo (MCMC). Multilevel or hierarchical approaches are scalable and easily parallelizable. These can be summarized by the following steps conducted iteratively: forming clusters within graph based on a suitable rule, collapsing the clusters into a single node (so that graphs are reduced to a manageable size

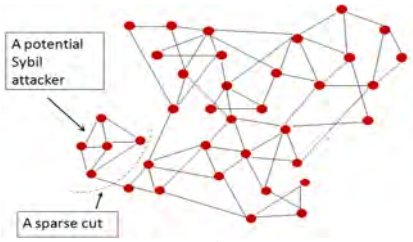

Fig. 1. Sparse cut of a weakly connected cluster.

for the next level), partitioning the coarsened graph, and uncoarsening the graph using refinement techniques such as [14]. The sparse cut problem differs from the min-cut problem in the sense that we do not aim to have partitions of equal size. The sparse cut problem differs from the k-way partitioning problem in that the number of partitions is not fixed. Moreover, for our problem, each peer node is a computing unit in itself and hence we employ a distributed decision making process.

C. Police Nodes

The presence of trusted nodes or super peers in the reputation graph is required in most existing referral frameworks. A verifier node which is honest is implicitly assumed in [21] and [8]. According to [5], the presence of trusted nodes makes the reputation system asymmetric and hence it can be made sybil-proof. In our graph, we assume the presence of trusted nodes which we call "police nodes". These could be specially embedded nodes purely for the purpose of sybil defense or nodes which have a long history of being trustworthy and helpful. A police node keeps track of trust relations between mostly "regular" nodes in its close vicinity of the trust graph. Hence it has a limited local view of the trust graph. It can acquire the information about the trust links from other nodes in its vicinity. It can also infer trust relationships by monitoring traffic it relays. We assume that police nodes are dense in the graph and well connected to each other by short geodesic (hop) distances, and every regular node is assumed to be only a short geodesic distance to some police node. Hence, police nodes occupy strategically important positions in the graph handling relatively heavy traffic loads, and can make rough inferences of social trust between its neighboring nodes by, *e.g.*, simply monitoring their associated traffic volumes. Alternatively, regular peer nodes may communicate their direct trust vectors to their local police node for purposes of referral. A police node can use such data to create a complete picture of its local trust graph and check it for consistency.

D. Detecting sybils trying to evade

There are two ways in which a sybil node might try to mislead the police nodes: by lying about an attack edge or by lying about a sybil edge. The former case is easier to detect. The police node may request for reputation information and referral paths from each node in its vicinity and verify the consistency of the information provided by one node with that provided by the other, *i.e.*, through some kind of (centralized) consensus procedure. Since one of the nodes attached to an attack edge is an honest node by definition, the sybil node

cannot mask or forge an attack edge. If the honest node attached to the attack edge is beyond the view of the police node then the lie will be detected at the next hierarchy of the algorithm when neighboring police nodes exchange information with each other, *cf.* Section IV.

Detecting a lie about a sybil edge is trickier and we need to assume additional capabilities for the police nodes to monitor traffic. If there exists at least one honest node within the view of the police node which has been referred to one of the sybil nodes using a sybil edge, then a sybil node cannot lie about it, since the honest node will enlighten the police node about its presence. Here we assume that a sybil node will always try to mask a sybil edge to generate a fake sparse cut within the sybil cluster. On the other hand, forging a non-existing sybil edge will only harm the attacker by making the sybil cluster connect densely within itself and hence making the attack edges appear sparser thereby improving detection probability. If such an honest node does not exist, then the police node needs to rely on the trust relationships it estimates by monitoring traffic in its vicinity as described above.

III. Background of the Algorithm

A. The Hierarchical Framework

The aim of the following algorithm is to identify sybil clusters from a decentralized database of reputation/referral paths in a distributed fashion with minimum inter-node communication. It's a scalable and computationally efficient alternative compared to an approach involving data aggregation and centralized decision making. The algorithm works at incremental hierarchical levels, where at each level the following steps are taken:

1. Neighborhood discovery (and trust-link's monitoring for the first hierarchy)
2. Randomized clustering
3. Determining sparse cut within each cluster if such a cut exists
4. Collapsing each partition (or the whole cluster if a sparse cut is not detected) to a single node
5. Incrementing the hierarchy level index

Incrementing the hierarchy implies considering the "collapsed" partition as a single node and the aggregated inter-partition edges as inter node edges. After collapsing the two partitions in a cluster, the police node in charge of the cluster represents both the nodes formed after the collapse.

B. Penalty for hierarchical sparse cut computation

Hierarchical sparse cut computation is clearly different from a centralized approach and might incur a "penalty" (typically an underestimation error) in terms of the sparse cut identified. This is due to the limited view that the nodes have and that cuts in the collapsed nodes may not share common end points for splicing. However, even when the sparse cut detection algorithm is centralized, heuristic methods are often employed. Our focus here is not a precise calculation for the whole graph, rather a scalable approximation that is useful for identifying sybil identities. The final outcome of our algorithm depends on the clustering performed at each hierarchy. For this reason, we randomized clustering in our algorithm when evaluating its performance.

IV. The Algorithm

A. Randomized Clustering

The algorithm to find a sparse cut in a graph is a heuristic which might not find a sparse cut even after multiple trials. However, the probability of identifying such sparse cut is quite good. The cluster heads wait for a random time before contacting nodes in successive hops. Note that our clustering objective is not to simply group similar objects together, rather we are interested in identifying sparse cuts within clusters that we suspected to be attack edges.

In randomized clustering, each node randomly decides to become a cluster head randomly (except at the first hierarchy where all the police nodes are the cluster heads). Each cluster head tries to rope in nodes in its vicinity in successive hops starting with directly connected one hop nodes. A police node queries a node it wishes to rope-in to find whether it has already joined a different cluster. If so, the requested node replies to the requesting node with the id of its cluster head. In this way the cluster head learns about its neighboring cluster heads. The cluster head waits for a random amount of time before it contacts more distant nodes. The expected value of this random waiting time increases after each hop. This ensures that the cluster size does not become too large. The cluster head continues to expand its cluster until it gets negative responses from all the nodes at that distance.

In order to limit the size of the clusters at higher hierarchies each cluster head might keep a track of the number of nodes it has grouped so far. When it encounters another cluster, it might also compete with neighboring cluster head based upon the size of its cluster.

B. Sparse Cut Computation and Partition Collapse

Min-cut determination is a well known problem and is closely linked with finding maximum flow in a graph. The heuristic [12] is simpler than Ford-Fulkerson [6] and can be easily modified for our purposes to find a "sparse" cut. Each cluster-head tries to find a sparse cut within its cluster using the algorithm originally presented in [12], called there the "contraction" algorithm, where it is shown that by collapsing two nodes along a randomly selected edge until only two clusters remain, a min-cut with high probability will be found after $O(n^2)$ trials where n is the number of vertices in the graph. Since the number of nodes in each cluster is limited in our case, the number of iterations required to find a sparse cut should be fairly small. For a sufficiently small cluster, we can also choose to do a brute-force cut search. If the sparseness of the min-cut is less than a threshold value then two partitions are formed within the cluster, else the complete cluster is considered as a single partition. Again, each element of the partition is collapsed to a single node for the next hierarchy and the police node represents the partitions formed from its cluster.

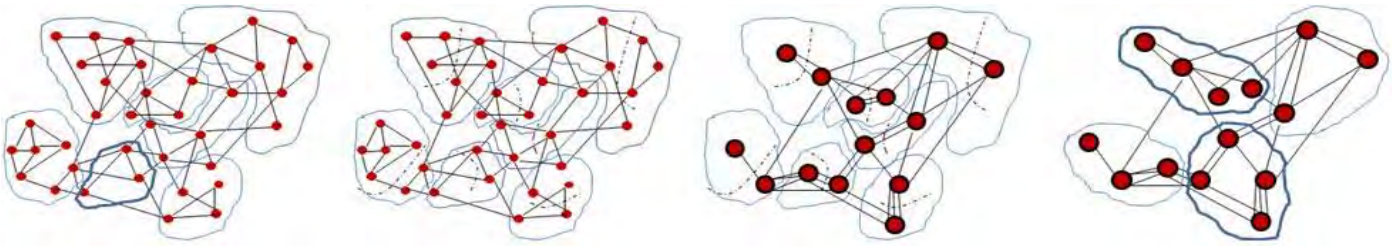

Fig. 2. Hierarchical sparse cut detection algorithm: Starting with randomized clustering, inter cluster sparse cut detection, partition collapse, incrementing the hierarchy and randomized clustering at the next hierarchy.

C. Incrementing Hierarchy

As nodes in a partition are collapsed into a single aggregate node, the new node thus formed has a weight equal to the sum of the weights of all aggregated nodes. We assign weight one to all the nodes at the first hierarchy. The weight of the edge between the new nodes formed in a given direction is equal to the combined weight of all the edges between the collapsed partitions in that direction. Maintaining node and edge weights this way allows sparse cuts to better approximate the corresponding operations on the lowest level graph. After incrementing the hierarchy, the graph becomes coarser and we repeat all the steps beginning with randomized clustering at the new hierarchy.

D. Some Definitions and Analysis

An honest edge is one connecting two honest nodes or one from a sybil node to an honest node. A sybil edge is one connecting two sybil nodes. An attack edge is an edge from an honest node to a sybil node. In these definitions we assume the edge from a sybil node to an honest node to be honest because it represents the trust earned by the honest node in the eyes of the sybil node. A homogeneous partition is one containing all similar nodes (either all sybil nodes or all honest nodes). In a cluster of sybil nodes, we refer to those sybil nodes with attack edges directed towards them as legitimate sybil nodes. The sybil attacker can have more than one "legitimate" identity, *i.e.*, more than one identity to which attack edges are directed. Sybil identities without attack edges are called "pure".

If there is just one legitimate sybil identity, all sybil identities will belong to one cluster after the first iteration. This is because the cluster head that ropes in the first (legitimate) sybil will surely rope in the rest of the sybils (pure). Then, with a high probability, they will be cleanly grouped by sparse cut into one aggregated node. That is, this aggregated node will contain all sybil identities and will be represented by the police node which was the cluster head of the original cluster. As we ascend the hierarchy, this aggregated node will not be further aggregated as this value of sparse cut will remain intact due to the absence of any other edge connecting this aggregated node with the rest of the graph so that, at the end, there will be just two clusters: the sybils and everything else. Hence the detection of a sybil with one legitimate sybil node is straight forward to analyze. It depends only on finding the sparse cut during the first hierarchy.

Now suppose there are multiple legitimate sybil identities of the sybil attacker. In the first iteration, it is possible that the legitimate sybil identities along with pure sybils will be partitioned and grouped into different clusters, where pure sybil identities can only be found together with at least one legitimate sybil identity. Nevertheless, sparse cuts will isolate the larger sybil clouds after the first iteration, and these larger sybil clouds will be grouped together in subsequent iterations. Due to our assumption that the sybil cloud is separated by a sparse cut, we can argue that the probability of an attack edge belonging to the sparse cut in a given hierarchy, provided that it belongs to the sparse cut in the previous hierarchy, increases with the level of the hierarchy. Similarly, we can argue using our assumption about the dense connectivity of networks containing honest nodes; that the probability of an honest edge belonging to the sparse cut in a given hierarchy, provided that it belongs to the sparse cut in the previous hierarchy, decreases with the level of the hierarchy.

As compared to the case of one legitimate sybil identity, the attacker has a better chance of evading detection if it distributes the attack edges among a larger set of legitimate sybil identities. Diluted reputations per legitimate identity will, however, naturally result for the same honest transactional effort. This is a fundamental trade-off when a defense based on sparse cuts is deployed.

We now provide a simple calculation for the case of a single sybil attacker with more than one legitimate identity. To this end, we first *fix* the following quantities:

- the total number of pure sybil identities, N_{ps}
- the mean edge-degree of an honest node, D
- the number of honest nodes in an *iteration-1* cluster, N_h, containing one legitimate sybil identity
- the number of attack edges, A, of that legitimate sybil identity.

Let n_{ps} be the random number of pure sybil identities which are present in the *iteration-1* cluster together with one legitimate sybil identity. Assume n_{ps} is uniformly distributed on $[0,N_{ps}]$, as would be the case if the sybil attacker has two legitimate identities in total. Assume the degrees of the honest nodes are independent as would be the case in a branching process approximation of the honest portion of cluster [18], beginning say at the legitimate sybil identity. Let d_{min} be distributed as the minimum degree among honest nodes in this cluster and assume that the degree distributions of the honest nodes are geometrically distributed so that

$$\mathrm{E}\ (d_{\min}) = \frac{D}{N_h}$$

***Theorem*:** For this highly idealized model and for all sufficiently large A,

$$P(\frac{A}{n_{ps}} > d_{\min}) > \frac{A/N_{ps}}{D/N_h}$$

Remark: This is the probability that the sparse cut along the attack edges is greater than that isolating the minimum-degree honest node. So, this is a lower bound on the probability that the sparse-cut step of the first iteration will fail to isolate the sybils in this cluster.

Proof:

$$P(\frac{A}{n_{ps}} > d_{\min}) = P(n_{ps} < \frac{A}{d_{\min}})$$

$$= \sum_{\substack{i>0 \\ i<A}} P(n_{ps} < \frac{A}{i}) P(d_{\min} = i)$$

$$= \frac{A}{N_{ps}} \sum_{\substack{i>0 \\ i<A}} i^{-1} P(d_{\min} = i) \quad \text{since } n_{ps} \sim [0\ ,\ N_{ps}]$$

$$\approx \frac{A}{N_{ps}}\ \mathrm{E}\ (\frac{1}{d_{\min}}) \quad \text{for all } A \text{ sufficiently large}$$

$$> \frac{A/N_{ps}}{D/N_h} \quad \text{by Jensen's inequality . } \square$$

V. Simulations

Simulations were performed using NetLogo [17], a multi-agent programmable modeling environment. We programmed the behavior of a single node and observed the combined behavior of the network of nodes each operating concurrently. The trust graph simulated consisted of 225 honest nodes, each with degree with average 3 and varying from 1 to 6. An honest node connected to a distant honest node with positive probability, so that the graph became densely connected. Each honest edge connected an honest node with one of the other honest nodes selected randomly from a set of 50 closest honest nodes with a probability of 0.25. A sybil cluster of 25 nodes, which was densely connected within itself, was connected to this graph by a sparse number of attack edges. We randomly chose sybil nodes from among these to act as legitimate sybil nodes. The police nodes were randomly selected from the set of honest nodes. All the honest nodes were equally likely to be selected as police nodes in our simulations. The number of police nodes was fixed at 22. Note that mean cluster size is inversely proportional to the number of police nodes.

In the first iteration, these police nodes performed randomized clustering and roped in as many nodes in their vicinity as they could into their cluster. Each cluster head (police node) identified a sparse cut within its cluster using the contraction algorithm. Again, according to [12], $O(n^2)$ trials are sufficient to determine a sparse cut. Hence we fixed the number of trials to 1000 given that the number of nodes in a cluster should include on average 10 honest nodes. We observed through our simulations that this value of number-of-trials for the contraction algorithm, for determining the intra-cluster sparse cut, was indeed sufficient.

For the 250-node graph, a 3-level hierarchy was appropriate because clusters formed by a factor of 6 to 1 after the first iteration. Again, the outcome of the complete run was considered successful if, after the final hierarchical cut, all the sybil nodes were placed in one partition and all the honest nodes in the other.

Fig. 3 shows the average probability of detecting a cut in a single 3-level hierarchy trial plotted against the number of attack edges. We kept the number of legitimate sybil nodes constant at 2. We varied the number of attack edges from 2 to 8. Every attack edge connected an honest node to a sybil node which was randomly chosen from the two legitimate sybil nodes. We ran an independent 3-level hierarchy trial of our algorithm for sybil detection on 300 different graphs generated randomly as described above. So, the sample standard deviation (*i.e.*, statistical confidence) of the estimates is approximately $((p\text{-}p^2)/300)^{0.5}$, where p is an average probability reported in Fig. 3. We observed the decrease in probability of success as the number of attack edges increased, which was expected since as the number of attack edges increase, the cut in the graph becomes less sparse. In the same experiment, we kept track of the probability of an event "A" when a single honest node is isolated, instead of the sybil nodes, in a cluster at the first level of the hierarchy. As discussed in the theorem above, we observed an increasing trend in this probability as we increased the number of attack edges.

Fig. 4 shows the result of a similar exercise where we instead varied the number of legitimate sybil nodes from 1 to 4. We kept the number of attack edges constant at 4. Every legitimate sybil node had at least one attack edge with an honest node. We ran a 3-level hierarchy trial of our algorithm on 100 different randomly generated graphs and found the average probability of detection. The results showed that the sybil attacker might improve its chances of evading detection by distributing the trust relationship it has with honest nodes among different sybil identities. But, again, this comes at the cost of diluting its accrued reputation as observed by honest nodes. Hence there is a tradeoff between distributing the attack edges between different legitimate sybil nodes to evade detection and the amount of reputation the legitimate sybil node can accrue for itself for transactional and referral benefit.

In our next experiment we varied the small-world property of the trust graph in the absence of a sybil attacker. We did this by varying the probability of connecting to a distant node. As this probability was increased, the graph became more densely connected. We ran our sparse-cut detection algorithm on this graph and observed the percentage of false positives. As observed in Fig. 5, the design of the sparsity threshold value is critical in minimizing the false positives and its calibration should take into consideration the measure of the small-world property of the trust graph consisting of only honest nodes.

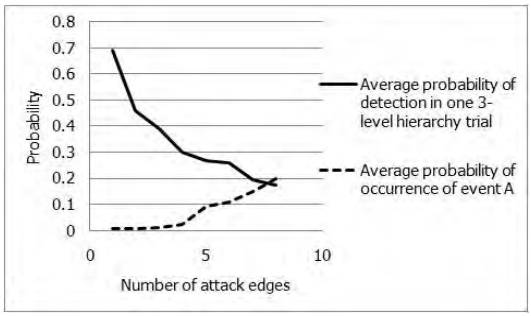

Fig. 3. The effect of number of attack edges on detection probability. The number of legitimate sybil nodes is constant at 2.

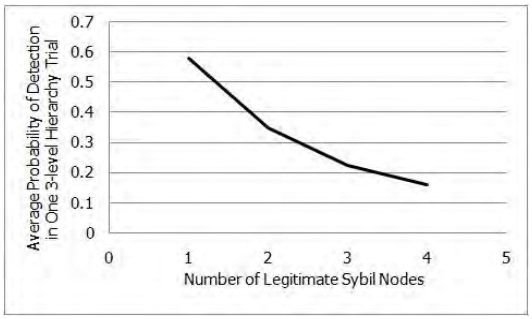

Fig. 4. The effect of number of legitimate sybil nodes on the detection probability. The number of attack edges is constant at 4.

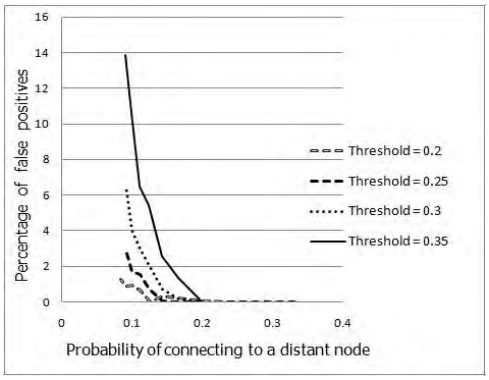

Fig. 5. False positives for different sparsity threshold values on graphs with varying small-world property.

VI. Summary and Future Work

A scalable and distributed hierarchical algorithm for identifying sybil clusters leveraging trusted police nodes is proposed along with measures to counteract the misleading actions by a sybil attacker. Sybil clusters are correctly identified when, after the final hierarchical computation, the sparse cut edge-set equals the attack edge-set. Every sparse cut computation is performed on a relatively small number of nodes in a distributed manner. We explored by simulation the trade-offs between false positives and missed detection depending on the chosen threshold for computed sparse-cut and on the social network topology.

In future work, one can explore the effect of other parameters, *e.g.*, those governing the topology of the graph of the honest nodes, and the effect of running the 3-level hierarchy multiple times, on the calibration of sparse-cut decision threshold trading off false positives with missed detection performance. Also, one can extend this algorithm to operate in an adaptive manner on dynamic graphs with peer churn and changing direct reputation values. Finally, distributed techniques can be employed to improve fault tolerance, to protect against otherwise misbehaving police nodes (byzantine attacks) [1], [2], and to detect multiple different attackers simultaneously.

References

[1] J. Aspnes. Randomized protocols for asynchronous consensus. *Distributed Computing* **16**(2-3): pp. 165-175, Sept. 2003.

[2] H. Attiya and J. Welch. Distributed Computing: Fundamentals, Simulations, and Advanced Topics. Second Edition. Wiley, 2004.

[3] R. Bazzi and G. Konjevod, ‘‘On the establishment of distinct identities in overlay networks,’’ in *Proc. ACM Symp. Principles of Distributed Computing (PODC)*, Las Vegas, NV, pp. 312–320, July 2005.

[4] P. Buser, ‘‘A note on the isoperimetric constant,’’ *Ann. Sci. cole Norm,* Sup.(4)15, no.2, 213–230, 1982.

[5] A. Cheng and E. Friedman, “Sybilproof reputation mechanisms,” *in Proc. ACM SIGCOMM Workshop on Economics of Peer-to-Peer Systems (P2PECON), Philadelphia, PA,* pp. 128–132, Aug. 2005.

[6] T. Cormen, C. Leiserson, R. Rivest, and C. Stein. Introduction to Algorithms, Second Edition. MIT Press, 2001.

[7] E. Damiani, D. C. di Vimercati, S. Paraboschi, P. Samarati, and F. Violante, “A reputation-based approach for choosing reliable resources in peer-to-peer networks,’’ *in Proc. ACM CCS*, 2002.

[8] G. Danezis and P. Mittal, “SybilInfer: Detecting sybil nodes using social networks,” *in Proc. NDSS, San Diego, CA*, Feb. 2009.

[9] M. Dell’Amico and Y. Roudier, “A Measurement of Mixing Time in Social Networks,” *in Proc. 5th International Workshop on Security and Trust Management, Saint Malo, France*, Sept. 2009.

[10] J. R. Douceur, ‘‘The sybil attack,’’ *in Proc. Int’l Workshop on Peer-to-Peer Systems (IPTPS)*, pp. 251–260, 2002.

[11] S. Kamvar, M. Schlosser, and H. Garcia-Molina, “The EigenTrust Algorithm for Reputation Management in P2P Networks,” *in Proc. ACM World Wide Web Conf.*, May 2003.

[12] D.R. Karger, “Global Min-cuts in RNC, and Other Ramifications of a Simple Min-Cut Algorithm” *in Proc. ACM/SIAM Symp. Discrete Algorithms*, pp. 21-30, 1993.

[13] G. Karypis and V. Kumar, “A fast and high quality multilevel scheme for partitioning irregular graphs,” *Technical Report TR 95-035, Department of Computer Science, University of Minnesota*, 1995.

[14] B. W. Kernighan and S. Lin, “An efficient heuristic procedure for partitioning graphs”, *The Bell System Technical Journal*, 1970.

[15] G. Kesidis, A. Tangpong and C. Griffin, “A sybil-proof referral system based on multiplicative reputation chains,” *IEEE Comm. Letters*, pp. 862-864, Nov. 2009.

[16] Monien, B., R. Preis, and R. Diekmann, “Quality Matching and Local Improvement for Multilevel Graph-Partitioning.,” *Parallel Computing* 26(12): pp. 1605–1634, 2000.

[17] NetLogo itself: Wilensky, U. 1999. NetLogo. http://ccl.northwestern.edu/netlogo/. Center for Connected Learning and Computer-Based Modeling, Northwestern University. Evanston, IL.

[18] M.E.J. Newman, S.H. Strogatz, and D.J. Watts, “Random graphs with arbitrary degree distributions and their applications,” *Phys. Rev. E*, **64**, 2001.

[19] M. Spear, X. Lu, N. Matloff, and S. F. Wu, “Karmanet: Leveraging trusted social path to create judicious forwarders,” *in Proc. Future Information Networks, 2009. ICFIN*, pp. 218-223, 2009.

[20] L. von Ahn, M. Blum, N. J. Hopper, and J. Langford, ‘‘CAPTCHA: Using hard AI problems for security,’’ *in Proc. Eurocrypt 2003*, Warsaw, Poland, pp. 294–311, May 2003.

[21] H. Yu, M. Kaminsky, P. B. Gibbons, and A. Flaxman, “SybilGuard: Defending Against Sybil Attacks via Social Networks,” *IEEE Trans. Networking*, **16**(3), pp. 576–589, June 2008.