SybilDefense: Defending Sybil attacks using techniques from Complex Systems

Kiran Kumar and Venkata Ponnam

Indiana University, School of Informatics and Computing,
Bloomington, Indiana
{knkumar,vponnam}@indiana.edu

Abstract. Web 2.0 applications and websites towards social news, gaming and networking are particularly vulnerable to Sybil attacks. In a Sybil attack, a malicious user assumes many fake identities, and pretends to be multiple honest users in the system. By controlling a certain fraction of the nodes in the system, he out votes the honest users to promote his own news articles, which poses a threat to the functionality of the original application.

Recent Sybil defense mechanisms are based on an assumption that malicious users might take several fake identities but few trust relationships. This assumption holds well for some peer to peer systems and distributed systems. However, recent studies on Facebook show sixty percent of users are willing to except friendship requests from unknown users. Furthermore, on Social news websites like Digg.com and career networking websites like Linkedin, a user is more willing to make connections with new and unknown people.

In this work, we explore both SybilDefense mechanisms at the level of recommendation algorithms that applications could use to make it hard for Sybil nodes to control an application despite out voting the honest users. We also look at SybilDefense mechanisms from the prior trust assumption, and compare it to existing defense techniques.

Key words: Sybil attack, Reputation System, Complex System, Social Networks, SybilDefense

1 Introduction

Web2.0 algorithms rely on Social interaction and participation rather than the machines themselves. For example, a traditional search engine algorithm would rank pages depending on the number of references to a web page. Advances in social networking and its utility now make it feasible to fine grain the ranking system by letting users vote or digg web pages that are of particular interest. A new set of social news websites like Digg.com, Reditt.com, del.ico.us are based on this idea. Such websites have high value as future search engines could buy and use data from these websites to better present search results that are of particular interests. On other other hand, existing social networks like Facebook are also widening their framework to collect such data, by including a like button.

However, such services that rely on its participating users are heavily vulnerable to Sybil attacks. In a Sybil attack, a user creates multiple fake identities, in way that he tries to out vote the number of participating honest users. Such a user can firstly destroy the functionality of the system. For example, a competing website could perform such an attack on its competitor to destroy its functionality, thereby gaining more users. On the other hand, a user performing a Sybil attack can promote third parties webpages, products or services on a website and make money. Such an attack is very common on Digg.com, were users are paid twenty five dollars to promote a website on to the first page of Digg.com.

Sybil attacks pose a real threat to the functionality and popularity of a web service. Most Internet services like Linkedin and Digg.com acknowledge this fact, and they try to defend against these attacks using trail and error mechanisms which often result in loss of money frequently. For example, Digg defended itself against these attacks by changing its reputation algorithm to accommodate for diverse and popular users. This is a good defense as it often takes a long time for a Sybil user to become popular, and the diversity makes it even more difficult. However, in this case, digg had to sacrifice some of its natural functionality. As the number of popular, diverse users present at a point of time varies and is often low, thereby popular news floats on its competitors websites like reditt before it appears on digg.

There are two forms of Sybil defenses, the centralized and the decentralized. Centralized defenses assume that the authority does some background verification when a user joins onto a network, or tries verifying the authenticity of a user based on his or her behavior after he starts participating on the network. Such a defense mechanism is used by Linkedin and by most web services. On the other hand decentralized mechanisms take many forms, where users block each other, or by studying the graph structures, or in our case by using algorithms from complex systems. However, it is a common belief that hybrids offer best defenses where algorithms can help to identify Sybil nodes and further verified manually used a centralized approach. This works well because most web based applications are owned by single companies. However, this also means loss of privacy to its participating users, but that domain is still an open problem and we do not address it in our work.

Sybil attacks are more formally studied both in industry and Academia. Sybilguard[2], Sybillimit[3], Sybilinfer[1] are most recently proposed defenses against these attacks. All these proposals assume that malicious users might take several fake identities but few trust relationships. This assumption holds well for some peer to peer systems and distributed systems. However, recent studies on Facebook show sixty percent of users are willing to except friendship requests from unknown users[]. Furthermore, on Social news websites like

Digg.com and career networking websites like Linkedin, a user is more willing to make connections with new and unknown people as participating users make use of pseudonyms.

Furthermore, Sybilguard[2], Sybillimit[3], Sybilinfer[1] work by taking the whole social network graph. They all try to identify the Sybil nodes by studying the social network graph. The intuition is Sybil nodes form less trust relationships, therefore are less connected to the core of the graph and often look isolated. Therefore, by taking a mincut of the graph it must be possible to identify these nodes. The fundamental idea is the same for all these defenses but they defer in the algorithms that are being used. For example, Sybilguard uses just a random walk whereas Sybilinfer uses a machine learning algorithm. Sybil limit further assumes that two thirds of the users are honest, which is unrealistic especially during the bootstrapping stage of a website, which is also a sensitive stage as the competitors might try to defame the website before it gets popular.

Furthermore, all these constructions rely on the fact that social networks exhibit Fastmixing, where the network balances itself over period of time and reaches a saturation stage after a while. This is well studied on few networks like facebook, but is not yet studied over networks where even the honest users use pseudo names to hide their identities for privacy gains. Also, if we were to assume a network is fast mixing it just means that the network cannot defend itself during the bootstrapping stage.

Our intuition behind this work is to use a complex systems algorithm which builds itself from simple rules and exhibits a complex behavior to defend against Sybil attacks, even when the prior assumptions don't hold. We successfully provide constructions that could defend against these attacks, even when networks don't exhibit fastmixing. Which means defense to Sybil attacks during their own bootstrapping phases. Therefore a new web service is likely to survive against it;s competitors if it uses our proposed algorithm.

2 Overview

The SybilDefense algorithm takes as an input a social graph $G = (V,\ E)$ as an input and a single known honest node that is part of this graph. And then we use trust relationships to distinguish between Sybil nodes and the honest nodes using ant clustering algorithm. Note that this will not require any assumptions that the network is fastmixing during the bootstrapping stage. Furthermore, our algorithm is based on features that are generic enough. Fore example, when dealing with facebook we could include features that are based on geographic location, similarity in interests with honest nodes and their connections. When on Linkedin we can use both the geographic locations in conjunction with the interests and professional establishments and connections with honest nodes in the systems that are well connected.The following conceptual steps are then applied to return the probability each node is honest or controlled by a Sybil attacker:

- A set of traces T are generated and stored by performing special random walks over the social graph G. These are the only information retained about the graph for the rest of the SybilDefense algorithm, and their generation is detailed in next section.
- A probabilistic model is then defined that describes the likelihood a trace T was generated by a specific honest set of nodes within G, called X. This model is based on our assumptions that social networks are fast mixing, while the transitions to dishonest regions are slow. Given the probabilistic model, the traces T and the set of honest nodes we are able to calculate $Pr[T|X\ is\ honest]$.
- Once the probabilistic model is defined, we use Bayes theorem to calculate for any set of nodes X and the generated trace T, the probability that X consists of honest nodes. Mathematically this quality is defined as $Pr[X\ is\ honest[T]]$.
- Since it is not possible to simply enumerate all subsets of nodes X of the graph G, we instead sample from the distribution of honest node sets X, to only get a few $X_0, ..., X_N$ $Pr[X\ is\ honest[T]]$. Using those representative sample sets of honest nodes, we can calculate the probability any node in the system is honest or dishonest.

The key technical challenge is making use of this distribution to extract the sought probability each node is honest or dishonest, that we achieve via sampling. We leave this for future work and concentrate our efforts towards techniques that could work against the bootstrapping assumption w.r.t the fastmixing property.

3 Model and algorithm

Here, we describe our complex systems approach to defend against Sybil attacks. We don't make any assumptions that the network exhibits fastmixing property. We assume that the Sybil nodes form less trust relationships w.r.t some features. In other words, we provide a further granularity w.r.t trust relationships via feature extraction compared to prior approaches disused in the introduction section.

Modified Ant colony clustering Identifying clusters in unlabeled data has been a continual problem for most clustering algorithms owing to differences in assumptions and contexts used for representation. Ideally, we should make few assumptions about the data and work with little prior information. This may not always be feasible and the need for more general clustering techniques arise[4]. Clustering for Sybil nodes can make use of many features of the network we are analyzing, observing fixed quotients and using them to guide the clustering

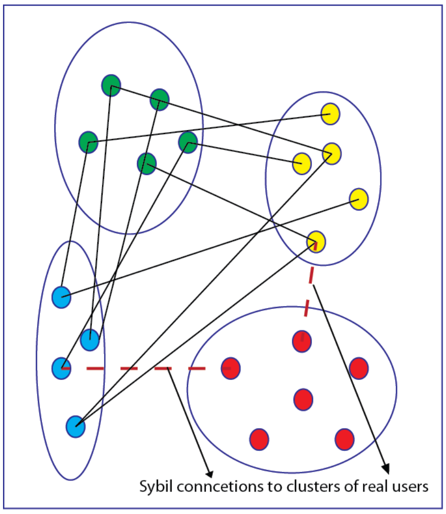

Fig. 1. Distinguishing Sybil nodes from Honest nodes using Ant Clustering

process.

Here we introduce the traditional ant clustering algorithm. The ants in nature use pheromones to communicate with other ants indirectly. Direct communication can happen through antennae contact. Ants have developed efficient protocols to separate food and dead ants in their colonies based on the indirect method of communication(using pheromones)[5]. Applying this process in computer science, we have artificial ants that can achieve distributed clustering.

Cluster Analysis based on ant clustering tends to identify common cluster maps and handles data efficiently. The main ideas of ant clustering is to leverage on a decentralized approach, and perform clustering in a subspace of the original dimension of the data. The ant colony organizes tasks in both a direct and indirect manner[7]. Traditionally ant clustering algorithms have been found to be efficient for clustering for not so good for topological mapping [6].

The standard ant clustering algorithm primarily uses the indirect approach based on stigmergy, but using only stigmergy causes its own problems. The standard Ant clustering algorithm[8] works by randomly scattering the data on a $2-D$ grid. The ants (agents) are then placed randomly at grid points which are empty, and are assigned a probability for picking up and dropping of the data based on the neighborhood density of similar structures. In practice this approach causes more clusters to be identified than actually present in the data[9]. We use a similar principle based on stigmergy, but instead of looking at the process as contained within a single ant colony, we use multiple ant colonies.

The algorithm can be divided into five phases:

Initialization The data we observe for networks in the real world are very high dimensional ,and clustering in such high dimensions proves costly and does not converge well[Chen]. The traditional ways to overcome this are to use a method for project the data to a lower dimensional space by using well known methods such as PCA. We suggest adopting a mechanism for dimensionality reduction based on the network model and the behavior characteristics in the network.

Scouting The number of colonies is decided randomly at the start. This algorithm encourages a moderately large number of colonies at the start. The scouts for a colony start at a random item in space and the mahalanobis distance between the item x_i and neighbor x_j is measured, if the distance is less than a threshold k_1 the item is added to the the colony.

$$f(x_i) = \begin{cases} C_j \Cup (x_i, x_j) & mahal(x_i, x_j) < k_1, x_j \in C_j \\ C_i \Cup (x_i, x_j) & mahal(x_i, x_j) < k_1, x_j \notin C_j \;\; Iftheneighborbelongstoanothercolony \\ \emptyset & otherwise \end{cases}$$

C$_j$, then the item is added to that colony C_j. If the neighbors do not belong to any colony, the item would be added to the colony of the ant C_i.

The probability that the data point is added to the cluster of the neighbor is given by:

$$P_{add} = \frac{k_1 + d_{mahal}}{k_1}$$

Cleaning Initial colonies formed may include some noise, because of slow convergence of colonies, and it needs to removed. The structure of the colonies may have changed and the data point may no longer belong to the colony, this also needs to be accounted for. The way we are handling this is by randomly sampling items that have been in the colony for a long time and checking if the item is still valid in the colony.

$$f(x_i) = \begin{cases} x_i \notin C_j & mahal(x_i, \sum_{c=0}^{n} x_c) > k_2, ph_j < k_p \\ \emptyset & otherwise \end{cases}$$

This is achieved by applying a pheromone evaporation ph to the item dropped to the colony. Every few iterations the items below a threshold pheromone level k_p are taken, matched with other items in the colony using mahalanobis distance and if they deviate from the colony structure are removed from the colony and reinitialized for other ants to process. The approach takes the view of a local clustering and considering the entire local cluster may produce better results.

The probability that the data point is removed from the cluster is given by:

$$P_{remove} = \frac{k_2 + \sum_i^j d_{mahal}}{k_2}$$

where i and j represent the boundary of the local neighborhood of that cluster.

Merging To achieve the optimal number of colonies without over-fitting, we merge the colonies if the number of common items exceeds a given threshold C_d. We check if the number of items is greater than a given threshold to allow for a good fit and avoid thrashing.

$$f(x_i) = \begin{cases} C_i \Cup C_j & HL(C_i, C_j) > K_m, mp_{ij} < mp_{th} \\ \emptyset & otherwise \end{cases}$$

The colonies are merged only if the Hellinger distance HL between the two data sets is less than the threshold for merging K_m. If the items are not similar the colonies are not merged and a memory retention parameter mp_{ij} is applied to remember the two colonies are different although they have common items.

New Colonies The number of colonies initially chosen may not represent the ideal set. As the ecology changes, new colonies may need to be formed. If there are items in the space which have been visited by agents from all colonies and have not been taken into any colony, a new colony is formed with the item.

$$f(x_n) = \begin{cases} xi \in C_n & x_i \notin \sum_{i=0}^{n} C_i \end{cases}$$

The colonies formed through the clustering approach would represent different human behavior patterns and comparing the connections among clusters would reveal the Sybil nodes. This could prove a efficient strategy to capture changing information in networks and Sybil behavior.

Preliminary results

Preliminary results with the iris data set has shown prominent cluster formation with 4 agents mapping the search space over 100 iterations. The clusters are not robust to noise. Common data points between 2 clusters tend to have a detrimental effort to clustering when marginally below the threshold for merge.

The merge itself performs well when the data points of the sampled cluster are compared with the cluster to be merged. Further effort needs to be enveloped on matching data points with cluster space for efficient de-evolution of the clusters.

The clustering itself is prone to noisy data, which is true of any network model. To establish robustness in the clustering process, we guide the user to choose characteristics of network data which would help identify a Sybil from legitimate user. Sybils tend to have a flood of requests to form new connections and most of then are rejected from legitimate users, this could form a good feature set to classify Sybil nodes. Other features such as common IP's and history of communication among users could also form good features to help classify Sybils.

4 Future work and Open problems

We presented an idea behind defending against Sybil attacks using hybrid models. Our algorithms rely on Complex systems approach, and do not require impractical assumptions like fastmixing at a boot scarping stage over social networks. This means that new web 2.0 applications could survive despite of existing competitors trying to put them down. However it remains open , whether or not decentralized systems can be provably efficient over some assumptions like finding mincuts in graph.

References

1. George Danezis and Prateek Mittal.: SybilInfer: Detecting Sybil Nodes using Social Networks. NDSS, February 2009.
2. Haifeng Yu, Michael Kaminsky, Phillip B. Gibbons, Abraham Flaxman.: SybilGuard: defending against sybil attacks via social networks. In Proceedings of the ACM SIGCOMM Conference on Computer Communications (SIGCOMM 2006), Pisa, Italy, September 2006.
3. Yu, H., Gibbons, P. B., Kaminsky, M., and Xiao, F.:SybilLimit: A Near-Optimal Social Network Defense against Sybil Attacks. In Proceedings of the 2008 IEEE Symposium on Security and Privacy.
4. Jain, A. K. and Murty, M. N. and Flynn, P. J. :Data clustering: a review. ACM Comput. Surv, Sept 1999.
5. Eric Bonabeau and Marco Dorigo and Guy Theraulaz:Swarm Intelligence: From Natural to Artificial Systems. J. Artificial Societies and Social Simulation. 2001.
6. Handl, J. and Knowles, J. and Dorigo, M: Ant-Based Clustering and Topographic Mapping. Artif. Life. Jan 2006.
7. Chen, Ke. On k-Median clustering in high dimensions. Proceedings of the seventeenth annual ACM-SIAM symposium on Discrete algorithm, SODA 2006.
8. Lumer,Erik D. and Faieta, Baldo. Diversity and adaptation in populations of clustering ants. Proceedings of the third international conference on Simulation of adaptive behavior. 1994.
9. Vito Trianni and Thomas Halva Labella and Marco Dorigo. Evolution of Direct Communication for a Swarm-bot Performing Hole Avoidance. ANTS Workshop. 2004.