UNIK: Unsupervised Social Network Spam Detection

Enhua Tan[1], Lei Guo[2], Songqing Chen[3], Xiaodong Zhang[2], and Yihong (Eric) Zhao[4]

[1]LinkedIn
entan@linkedin.com

[2]Ohio State University
{lguo,zhang}@cse.ohio-state.edu

[3]George Mason University
sqchen@cs.gmu.edu

[4]Yahoo! Inc.
yzhao@yahoo-inc.com

ABSTRACT

Social network spam increases explosively with the rapid development and wide usage of various social networks on the Internet. To timely detect spam in large social network sites, it is desirable to discover unsupervised schemes that can save the training cost of supervised schemes. In this work, we first show several limitations of existing unsupervised detection schemes. The main reason behind the limitations is that existing schemes heavily rely on spamming patterns that are constantly changing to avoid detection. Motivated by our observations, we first propose a sybil defense based spam detection scheme SD2 that remarkably outperforms existing schemes by taking the social network relationship into consideration. In order to make it highly robust in facing an increased level of spam attacks, we further design an unsupervised spam detection scheme, called UNIK. Instead of detecting spammers directly, UNIK works by deliberately removing non-spammers from the network, leveraging both the social graph and the user-link graph. The underpinning of UNIK is that while spammers constantly change their patterns to evade detection, non-spammers do not have to do so and thus have a relatively non-volatile pattern. UNIK has comparable performance to SD2 when it is applied to a large social network site, and outperforms SD2 significantly when the level of spam attacks increases. Based on detection results of UNIK, we further analyze several identified spam campaigns in this social network site. The result shows that different spammer clusters demonstrate distinct characteristics, implying the volatility of spamming patterns and the ability of UNIK to automatically extract spam signatures.

Categories and Subject Descriptors

C.2.0 [**Computer-Communication Networks**]: General—*Security and protection*

Keywords

Social networks; spam detection; community detection; user-link graph; social graph

1. INTRODUCTION

Spam in online social networks increases quickly because of the viral distribution of information provided by massive social connections on the Internet. The effectiveness of email spam detection also contributes to such a trend. A study [2] shows that email spam has dropped by half in 2010 and spammers are more aggressively targeting social networks and search engines. It is estimated that 67% of social network users have been spammed in a survey [1] conducted by Sophos.

To detect spam in online social networks, many supervised machine learning based methods have been proposed. For example, Lee et al. [9] proposed to deploy honeypots in social networks, and apply machine learning to detect spam using captured spam as the training set. Benevenuto et al. [3] suggested to detect promoters and spammers in a video social network with user-based, video-based, and social network based features. Markines et al. [11] proposed to use six features at post-, resource-, or user-level to capture spam in a social bookmarking system. However, supervised machine learning methods have some inherent limitations when being applied to a large social network site. Specifically, labeling the training set is required for supervised learning, which incurs a high human labor cost. Moreover, the labeling work has to be done repetitively to maintain effectiveness for spam detection given the volatility of the spam content and some spam posting patterns. Lastly, the supervised model always lags behind spam attacks with new patterns of spam content.

Different from supervised ones, unsupervised schemes that do not have the training cost have been proposed to detect spam in emails and social networks by directly leveraging the spamming patterns. For example, Xie et al. [19] proposed AutoRE to automatically extract spam URL patterns based on the distributed and bursty patterns of botnet-based email spam campaigns. Applying this approach in detecting social spam, however, may suffer from a high false negative rate since a number of spam posts in social networks are continuously sent over months instead of following the bursty pattern [13]. Gao et al. [6] identified spam by clustering posts based on text and URL similarities and then expecting spam posts to form large clusters with bursty posting patterns. This approach assumes that spam clusters are not connected to non-spam ones. However, spam posts may include non-spam URLs to increase their legitimacy as shown in our data and discussed in [19], which effectively connects spam clusters to non-spam clusters, making it highly difficult to distinguish spam from non-spam. Thereby, it is desirable and imperative to design an unsupervised scheme that can address the limitations of existing schemes.

In this work, we first propose a sybil defense based spam detection scheme SD2. In SD2, a user graph is constructed by combining the social graph and the user-link graph. The former represents the social relationship between active non-spammers, while the latter characterizes the spam link sharing activity of spammers. Observing that spammers and non-spammers usually form different communities in the user graph, SD2 applies community detection based sybil defense algorithm to the user graph and achieves better spam detection performance than existing schemes. However, because the effectiveness of sybil defense is subject to the spam attack intensity, SD2 does not perform well when the level of attacks increases.

To improve the spam detection performance under an increased level of spam attacks, we further design a new UNsupervised socIal networK spam detection scheme, called UNIK. Instead of picking out spam directly, UNIK works by capturing the properties of non-spammers in the network first, and then clustering suspicious spammers based on the landing pages they are advertising, leveraging both the social graph and the user-link graph. The underpinning of UNIK is that while spammers constantly change their patterns to evade detection, non-spammers do not have to do so and thus have a relatively non-volatile pattern. UNIK first constructs a user-link graph connecting users who share URL links. Given that a spammer often uses different accounts to post spam URLs, the user-link graph constructed would include almost all spammers in the system, although non-spammers who share URLs are also included. UNIK then constructs the social graph according to the mutual social connections between users, and identifies non-spammers with the help of the social graph. The URLs mainly posted by these identified non-spammers are collected as a URL whitelist, which captures the patterns of non-spam URLs. By trimming non-spam URL edges matching the whitelist in the user-link graph, UNIK isolates a large portion of non-spammers in the user-link graph. Finally, UNIK differentiates spammers from non-spammers with respect to the node degree in the trimmed user-link graph and detects the majority of spammers.

UNIK is expected to overcome the limitations of two existing unsupervised spam detection schemes [19], [6] and SD2, as UNIK exploits non-spam patterns to detect spam. First, the AutoRE scheme [19] works by detecting spam that is sent with two patterns: distributed and bursty. Correspondingly, spam that is typically posted in the same long duration as normal posts will not be detected. UNIK works by removing non-spammers from the user-link graph which covers most spammers, so it is able to detect most of the spam. Second, the spam clustering scheme [6] relies on the assumption that spam and non-spam posts can be clustered into different groups utilizing the sharing URLs between them. However, as shown in [19], spam content often includes non-spam URLs to increase the legitimacy, which effectively breaks the assumption even if only a handful legitimate URLs are included. UNIK overcomes this limitation by using the non-spam pattern to remove non-spam URLs from the user-link graph, therefore it is robust to the spam attacks with legitimate URLs. Third, SD2 uses a sybil defense algorithm to cluster non-spammers and spammers, whose performance is subject to the spam attack intensity. UNIK identifies non-spam URL signatures based on the social graph and the URL sharing pattern, and then removes non-spam URLs from the user-link graph, thus its effectiveness is maintained in spite of the spam attack intensity since non-spam patterns are largely not affected.

We evaluate the performance of SD2 and UNIK with a 10-month dataset from a commercial social blog site. SD2 shows its superior performance compared to AutoRE [19] and the spam clustering scheme [6] by reducing both the false positive rate and the false negative rate in detecting spam. UNIK also shows comparable performance to SD2 when being applied to the dataset. Furthermore, UNIK maintains its performance when the spam attack increases, while the performance of SD2 degrades accordingly.

Based on the spam detection result of UNIK, we have identified a number of large spam campaigns in this social blog dataset. Our analysis shows that different spam campaigns demonstrate distinct characteristics. On one hand, this indicates the ineffectiveness of some detection schemes relying on these spamming patterns. On the other hand, this also means that UNIK is able to effectively group spammers into spam campaigns, which provides an opportunity to extract spam signatures from these campaigns and use them to detect spam in other systems.

The rest of the paper is organized as follows. Section 2 evaluates the limitations of existing work and motivates our work. Section 3 presents the design, evaluation and analysis of SD2. Section 4 illustrates the design of UNIK, and Section 5 evaluates its performance. Section 6 analyzes the spammer clusters detected by UNIK. Section 7 discusses other related work, and Section 8 concludes this paper.

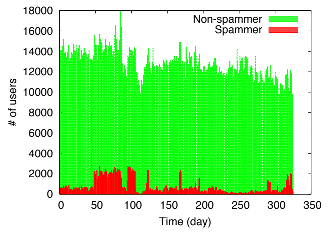
2. MOTIVATION

In this section we present the limitations of existing unsupervised spam detection schemes based on a dataset from a large social network site. The results motivated our new designs in this study.

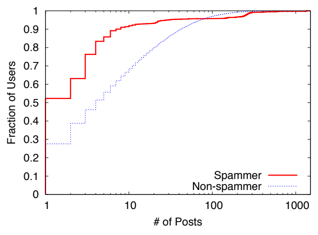
2.1 Dataset

We have collected posts and social connections of users for over 10 months from a large commercial social blog site in 2009 [16]. The number of user IDs who have at least one URL in their posts is more than 176 thousands. These IDs have more than 2 million posts with URL(s) in the trace collection duration. We developed a supervised machine learning algorithm to detect spammers in this dataset. The spammers detected by the supervised algorithm have both the false positive rate and the false negative rate around 1% to 2%. Because of the accuracy of the supervised algorithm and the infeasibility to label every user in the dataset, we use the results of the supervised algorithm as a ground truth in evaluating the performance of unsupervised spam detection schemes. As we have aforementioned, the good performance of a supervised algorithm is achieved with a price, thus it is still desirable to discover an unsupervised algorithm with similar performance.

Figure 1(a) shows the number of new user accounts created every day, stacked with non-spammers on top of spammers. Although the number of spammers is relatively small, some of the spammers posted a number of spam articles to the site as shown in Figure 1(b).



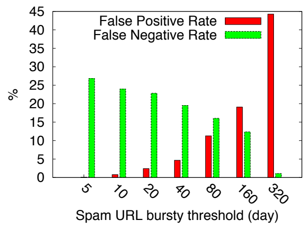
(a) New users created every day



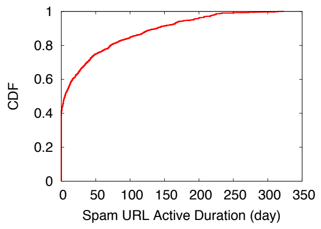
(b) Cumulative distribution of post number by spammers/non-spammers

Figure 1: Statistics of a social blog dataset

2.2 Evaluation of AutoRE



(a) AutoRE performance



(b) Active duration of spam URLs

Figure 2: Evaluating AutoRE

We first discuss the performance of AutoRE [19] that detects email spam with distributed and bursty patterns. We applied AutoRE to our social blog dataset by replacing the AS number with the user ID as hinted by [6]. Figure 2(a) shows that the suggested 5-day threshold of spam URL active duration has a false negative rate of 26.8%. We further changed the threshold from 5 days to 320 days. As a result, most spam is detected, but the false positive rate is increased to 44.3%. Figure 2(a) shows the performance results of AutoRE by tuning the threshold between 5 days and 320 days. Clearly, tuning the threshold cannot help to improve the overall performance.

AutoRE suggests that most spam URLs have a bursty active duration. However, Figure 2(b) shows that in our dataset, more than 50% of spam URLs have an active duration of more than 5 days. This finding indicates that if we rely on the bursty pattern of spamming to detect spam, we risk to miss a significant portion of spam that is as active as non-spam [17].

Our study shows that the main reason for the performance degradation of AutoRE is due to the change of the spam pattern: lots of spam has been posted with a non-bursty pattern instead of a bursty one.

2.3 Evaluation of FBCluster

Gao et al. [6] proposed a scheme, referred to as FBCluster, to detect spam clusters in a social network site. FBCluster constructs a similarity graph of posts on which posts sharing the same URL or similar text are connected. Spam clusters in the graph are detected if a cluster is distributed and bursty, i.e., the cluster is posted by more than 5 user IDs and the median interval of posts is less than 1.5 hours as suggested by FBCluster.

According to the FBCluster scheme, we first construct a URL sharing graph connecting posts sharing the same URL in our dataset. Then we apply the suggested thresholds (5, 1.5 hrs) to detect spam, and we get a false positive rate of 39.3% with a false negative rate of 0.2%. The low false negative rate suggests that FBCluster is able to detect most spam in our dataset as spam URLs are often shared by spam posts. However, the high false positive rate indicates that FBCluster has mistakenly included a number of non-spam posts in the detected spam clusters. We examine the largest cluster in our dataset, and find it has a non-trivial percentage (14.4%) of non-spam posts. Even if we only select the largest cluster as the detected spam cluster, the false positive rate is still as high as 30.1% while the false negative rate increases to 7.0%.

To understand why FBCluster has such a high false positive rate, we further check the URLs that are posted by both spam and non-spam posts. We find there are about 0.7% of non-spam URLs that are also presented in spam posts. We call this phenomenon the *legitimate URL inclusion attack*, as spammers include legitimate URLs in their posts to avoid being detected [19]. Although the scale of this attack is small in the dataset, any occurrence of such attacks will effectively connect a spam cluster and a non-spam cluster. Therefore, the reason for the high false positives is that FBCluster is unable to separate spam clusters from non-spam clusters when there exist legitimate URL inclusion attacks. This limitation of FBCluster is rooted from the fact that spammers can intentionally add legitimate URLs to their posts, increasing the difficulty of distinguishing spam from non-spam.

3. APPLYING SYBIL DEFENSE

3.1 SD2: Sybil Defense based Spam Detection

In the previous section, we have shown the limitations of existing unsupervised schemes in detecting spam. We notice that FBCluster [6] is able to detect the majority of spam in the trace, however, it fails to separate spam clusters from non-spam clusters, and leads to a high false positive rate. This phenomenon is very similar to the sybil attack, in which sybil nodes are connected with non-sybil nodes. This motivates us to investigate sybil defense based schemes [20, 4, 18] to detect spam, given such studies have not been conducted before.

However, directly applying sybil defense schemes for spam detection has several challenges. First, existing sybil defense schemes use the social network graph to identify non-sybil/sybil nodes. As a result, a non-trivial portion of low degree nodes need to be removed in the pre-processing [20, 4] of the social graph, in order to shorten the mixing time [12]. This prevents sybil defense schemes from detecting all the spam as a number of spammer IDs will be removed in the pre-processing. Second, spammer IDs are not necessarily participating in the social network at all because it is hard for spammers to convince non-spammers to be their friends.

Motivated by our findings, we propose SD2, a *Sybil Defense based Spam Detection* scheme by using the social graph and a user-link graph that connects users sharing the same URL. SD2 overcomes the problem of FBCluster by effectively separating non-spammers from spammers with the sybil defense scheme. SD2 also includes most spammers for detection by capturing the intensive URL sharing activities among spammers with the user-link graph to make the best use of the sybil defense scheme. Ideally, SD2 only removes inactive/new non-spammers and few spammers in the pre-processing, and then detects non-sybil nodes as the non-spammers and sybil nodes as the spammers, resulting few false positives and few false negatives.

In general, SD2 works as follows. First, a social graph connecting users with mutual social connections is constructed. Second, a user-link graph connecting users sharing URLs is added to the social graph, generating a user graph including almost all users we are interested in. Third, a preprocessing procedure is applied to remove nodes with less than 3 degrees [4]. Fourth, community detection based sybil defense [18] is applied to the user graph to rank nodes with the expectation that non-sybil nodes have higher ranks and sybil nodes have lower ranks. Finally, a cutoff is applied to ranked nodes to identify sybil nodes as spammers.

In SD2, a critical step is to find out the right cutoff point, for which we propose a method based on conductance ratio. For a set of nodes A in a graph, *conductance* [10] reflects the community structure of A. Define $B = \bar{A}$, or the complement of A, then conductance is the number of edges between A and B, divided by the sum of node degrees in A (or B if the number is smaller). If we define e_{AB} as the number of edges between A and B, e_{AA} (e_{BB}) as the number of edges within A (B). Then conductance of A is defined as:

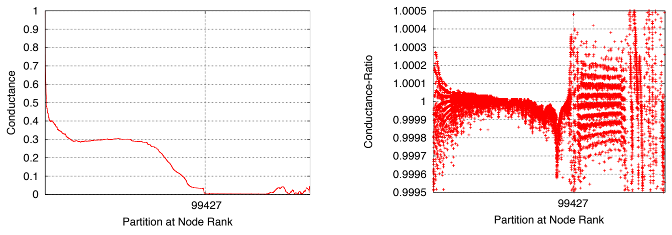
$$conductance(A) = \frac{e_{AB}}{e_{AB} + 2 \times min(e_{AA}, e_{BB})}$$

Clearly, conductance indicates the intensity of connections between A and the rest of the graph B. Conductance of 0 means strongest community structure (no outside connections), while 1 means weakest community structure (all connections are external).

The community detection based sybil defense algorithm [18] outperforms existing sybil defense approaches by utilizing the conductance metric. The algorithm starts from a trust node or trust node set s. New neighbor nodes of s is repeatedly added to the s with the preference of minimizing conductance, until all connected nodes are added. The community detection algorithm ranks nodes by the order of adding to the trust node set s, as the nodes ranked higher are more likely to be non-sybil nodes.

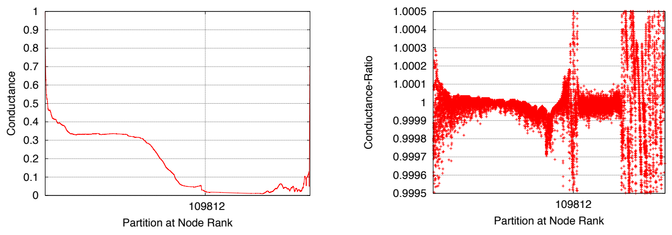
In applying the community detection based sybil defense algorithm, SD2 needs to find an optimal cutoff point to separate non-sybil nodes from sybil-nodes. However, there is no suggestion on the cutoff point selection from the algorithm. Therefore, we propose a cutoff method for SD2 based on our observations from the dataset. Figure 3 shows the conductance and conductance-ratio values of ranked nodes in the user graph constructed with the dataset. The conductance value is computed when the node is added to the trust node set. The *conductance-ratio* is computed as the ratio of new conductance to previous conductance upon adding a new node. As shown in Figure 3, the conductance-ratio is very close to 1 until it hits the middle part, where the value starts to vibrate sharply. SD2 thus selects the cutoff point upon the sudden increase of the conductance-ratio.

In terms of spam posts detection, SD2 outperforms existing schemes substantially as shown in Figure 9(b): the false positive rate is 2.8% and the false negative rate is 1.4%. If we consider the spammers detection performance, the false positive rate is 0.9% and the false negative rate is 3.0%.



(a) Conductance (b) Conductance-ratio

Figure 3: Conductance of the user graph (social graph + user-link graph) of original dataset



(a) Conductance (b) Conductance-ratio

Figure 4: Conductance of the user graph with simulated sybil attacks

The reason for SD2 to outperform existing schemes is twofold. First, it is able to effectively separate non-spammers from spammers by detecting the community of non-spammers with the social graph. Second, it is able to detect most spammers by including them with the user-link graph, even if most of them are not in the social graph.

3.2 Limitations of SD2

Although SD2 shows very good performance when it is evaluated with our real world dataset, it does have some limitations. Specifically, the performance of sybil defense in separating non-spammers from spammers degrades when the number of sybil attack increases, which results the degradation of SD2 performance.

Figure 4 shows the conductance and conductance-ratio of our dataset with simulated sybil attacks. We randomly select 10% of users in the social graph, and add the same number of spammers connecting to these users, forming a scale free social network among them. As a result, the cutoff point leads to higher false negative rate of 10.0% in terms of spammer detection performance.

4. DESIGN OF UNIK

Because the performance of sybil defense based unsupervised spam detection scheme SD2 degrades when there is an increasing level of attacks, in this section, we further design a new scheme UNIK: UNsupervised socIal networK spam detection. UNIK aims to overcome the limitations of SD2 by exploring the social graph and user-link graph separately.

4.1 Overview

Constructing the social graph is straightforward based on mutual relationships between any two users. In addition to that, in an online social network, users post content and

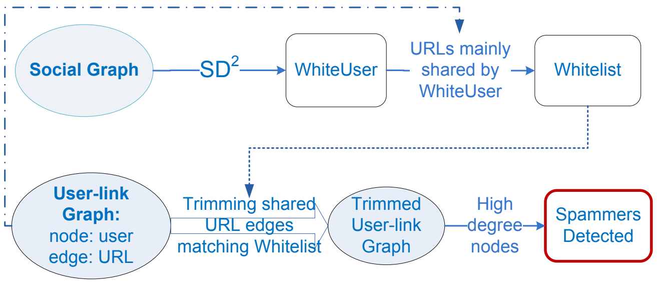

Social Graph
SD2
WhiteUser
URLs mainly shared by WhiteUser
Whitelist
User-link Graph: node: user edge: URL
Trimming shared URL edges matching Whitelist
Trimmed User-link Graph
High degree nodes
Spammers Detected

Figure 5: UNIK workflow

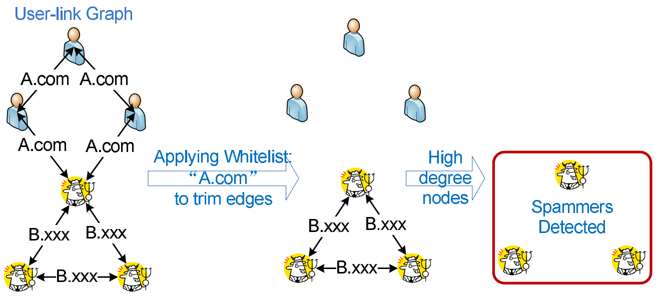


Figure 6: An example of edge trimming in UNIK

URLs. For spammers to promote some spam sites or advertisements, they keep posting the URLs pointing to the spam sites. And they often do this using different user accounts. Thus, UNIK also constructs a separate graph based on the posted URLs by each user in the network. On this graph, users are defined as nodes, and the shared URLs posted by them are defined as edges, and we call it a *user-link graph*.

At a high level, UNIK starts with a user-link graph based on the relationship between users and the URLs posted by them. Instead of directly detecting spammers, UNIK takes careful steps to remove non-spammers from this user-link graph with high confidence by leveraging both the social graph of social network users and the user-link graph based on the content. Because non-spammers do not have to constantly change their patterns as spammers do to escape from detection, non-spammers have relatively stable patterns. By deliberately removing non-spammers based on their patterns, UNIK is minimally affected by the sybil or legitimate URL inclusion attack by spammers, and achieves a better performance.

The underpinnings of UNIK are twofold. First, non-spammers are the main body of the social network (we assume that if spammers become dominant, the social network is no longer valuable to perform any detection). Second, spammers usually keep posting the same spam URLs with the same or different user accounts. Thus, spam URLs have high occurrences in the content on the social network.

Accordingly, UNIK works as follows. First, it constructs a user-link graph based on the posted URLs by the users, and a social graph based on the mutual relationship of the users. Second, it leverages the social graph to identify non-spammers. A URL whitelist is constructed by identifying the sharing activities of their posted URLs. Third, when user-link graph URL edges are filtered by the URL whitelist, UNIK makes most non-spammers become isolated or low-degree nodes on the user-link graph, which can be removed safely and left spammers node detectable. Figure 5 depicts the workflow of UNIK and Figure 6 shows an example on the edge trimming in separating non-spammers from spammers.

4.2 Generating Whitelist to Trim Edges

UNIK first tries to identify *non-spammers* by applying the SD2 algorithm to *the social graph only.* In contrast, the standalone SD2 algorithm detects *spammers* by using the combination of the social graph and the user-link graph, as the latter is required to connect spammers. Since non-spammers are mostly active in the social graph, we do not need to combine the two graphs in this step.

As shown in the last section, SD2 might have a non-trivial false negative rate in detecting spammers upon a sybil attack to the social graph. Therefore, the identified non-spammers are not 100% correct. Fortunately, UNIK manages to tolerate the error in the identified non-spammers list as shown in follows.

Based on the identified non-spammers list, UNIK generates a whitelist covering the URLs in identified non-spammers' posts, so that more non-spammers can be identified with the whitelist. However, if a spam URL is incorrectly included in the whitelist, the spammers sharing this spam URL may be removed from the user-link graph, causing false negatives. This situation is even worse for those widely shared URLs. To deal with such situations, UNIK uses the identified non-spammers and the user-link graph to help detect such spam URLs. For a URL shared among users, we have more confidence to include it on the whitelist if more than half of the users sharing this URL are non-spammers. That is, UNIK requires shared URLs to meet this condition to avoid inclusion of spam URLs in the whitelist. Note that the whitelist can be built based on domains or hosts, other than the URL itself, because a wider coverage of the whitelist is able to decrease false positives while errors in the whitelist only lead to the increase of false negatives.

Based on the generated whitelist, UNIK examines the edges on the user-link graph. Shared URL edges in the user-link graph are trimmed if they match the whitelist. After these removals, non-spammers who only share whitelisted URLs become isolated on the user-link graph because all their edges are trimmed. Thus, they can be removed, and the remaining nodes on the user-link graph are mostly spammers, with a limited number of non-spammers whose edges are mostly trimmed.

4.3 Applying the Threshold of Node Degree

The trimmed user-link graph may include some non-spammers because the whitelist is not likely to cover every non-spam URL. To further improve the detection performance, UNIK aims to remove as many non-spammers as possible based on the URL sharing properties of spammers. Typically, the URL sharing activities of spammers are much more intensive than non-spammers in terms of the number of shared URLs or the number of sharing users. This means that a spammer node often has more edges than a non-spammer node in the user-link graph. UNIK thus examines the node degree, and detects users whose degree is beyond a threshold as spammers.

To compute the node degree on the trimmed user-link graph, intuitively, edge(s) should exist between every two users sharing URL(s), and the edge weight is set to 1. This however has a time complexity of $O(n^2)$ as all the users sharing a URL are fully connected. To reduce the processing time, instead, for each shared URL, UNIK organizes the users sharing the URL into a circular node list, and only adds two edges to a node, one connecting the node to its preceding node and the other connecting it to its succeeding node, with each edge weight set as half of the number of other sharing users. By the increase of the edge weight, for each shared URL, the sum of edge weight of a node increases with the same amount as in a fully connected graph, which is the number of other sharing users. In this way, the user-link graph is quickly constructed with a linear number of edges connecting nodes, while the sum of edge weights of each node $\sum_{shared\text{-}URLs} other\text{-}sharing\text{-}users$ is exactly the same as the node degree in a fully connected graph $\sum_{other\text{-}sharing\text{-}users} shared\text{-}URLs$. With the edge weight, UNIK applies a heuristic threshold on the sum of edge weights, below which the connected nodes are deemed as non-spammers and get removed. In the end, only spammers exist on the user-link graph, possibly with few non-spammers who have intensive URL sharing activities as well.

The threshold of the node degree or the sum of edge weights needs to be determined beforehand for UNIK to work. In the next section, we will show that such a threshold can be independently determined (Figure 7(b)). For the best spam detection result, this threshold can be estimated by gauging a small sample of spammers comparing to non-spammers from time to time.

Algorithm 1 UNIK spam detection

```
Input: user_link_graph: (V, E) where V are users and E are
  shared URLs between V (linearly added)
Input: social_graph: (V, E) where V are users and E are social
  connections
Input: edge_weight_sum threshold $w$
Output: spammers_detected (spam_detected are the posts by
  spammers_detected)
  for v in social_graph.V do
    if v.degree $<$ 3 then
      remove v from social_graph.V
    end if
  end for
  WhiteUser $\leftarrow$ social_graph.V - SD2(social_graph)
  whitelist $\leftarrow$ set()
  for v in WhiteUser do
    for url in v.posts do
      if url not in whitelist and len(sharing_users(url) in
      WhiteUser) >= len(sharing_users(url)) * 0.5 then
        whitelist.add(url)
      end if
    end for
  end for
  for e in user_link_graph.E do
    if e matches whitelist then
      remove e from user_link_graph.E
    end if
  end for
  for v in user_link_graph.V do
    if v has no edges then
      remove v from user_link_graph.V
    end if
  end for
  spammers_detected $\leftarrow$ list()
  for v in user_link_graph.V do
    if sum_edge_weights(v) $>$ $w$ then
      spammers_detected.append(v)
    end if
  end for
```

The UNIK spam detection algorithm is shown by Algorithm 1.

5. PERFORMANCE EVALUATION

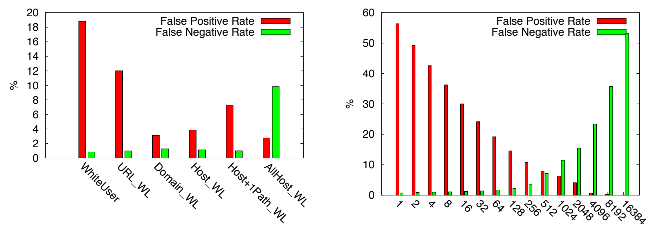
We have evaluated the unsupervised spam detection scheme UNIK with our dataset shown in Section 2.1. Table 1 shows the statistics of the social graph and user-link graph (constructed with a linear number of edges) built with the dataset.

Table 1: Statistics of social blog dataset

Type	Nodes	Edges	Spammers
Social Graph	141,619	474,701	206
User-link Graph	176,692	578,862	79,344

As UNIK applies whitelist and edge-weight-sum threshold to detect spammers, both of which can have different choices and both of which can be used independently, we first evaluate their different choices individually in order to find the best suitable ones.

5.1 Evaluation of Whitelist and Edge-weight-sum Threshold



(a) Applying different types of whitelist only (b) Applying different thresholds of edge-weight-sum only

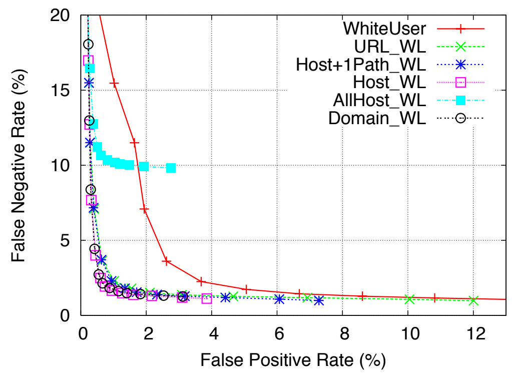
Figure 7: Whitelist and edge-weight-sum threshold evaluation

We first evaluate the spammer detection effectiveness by only applying different types of whitelist, including URL-based, host-based, and domain-based whitelist. We also evaluate `Host+1Path` based whitelist, where the whitelist matches the hostname and the first path in the URL. For example, if the whitelist pattern is `http://a.com/service`, then URLs like `http://a.com/service/1.html` or `http://a.com/service?q=1` will be matched. `Host+1Path` whitelist is a hybrid between URL-based and host-based whitelist. Lastly, we evaluate the approach of not using whitelist to trim edges, but only removing identified non-spammer nodes, namely the *WhiteUser* approach.

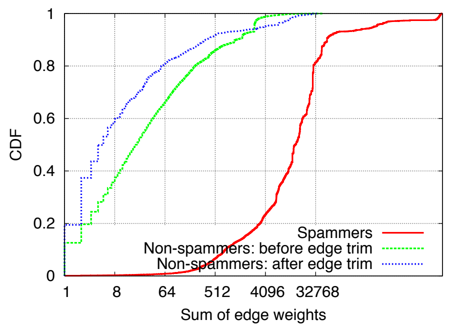
Figure 7(a) shows the spammer detection results after applying the whitelist or WhiteUser only. Because WhiteUser only covers identified non-spammers and the URL-based whitelist has the narrowest coverage of non-spam URLs, the user-link graph still has a number of non-spammers remaining, which results the highest false positive rate, but the lowest false negative rate as most spammers are still remaining. The domain-based whitelist and the host-based whitelist further decrease the false positive rate, since the coverage of non-spammers increases. However, any errors in the whitelist may result in a higher false negative rate as the spammers will be removed from the graph. The Host+1Path based whitelist outperforms the URL-based whitelist in terms of the false positive rate, and its false negative rate is also smaller than that of the host-based whitelist. Figure 7(a) also shows that if we include all URL hosts of identified non-spammers in the whitelist (all-host-based whitelist), the false negative rate increases substantially. This indicates that it is necessary to check whether the URL is mainly shared among identified non-spammers to avoid the inclusion of spam link in the whitelist.

We then evaluate applying the edge-weight-sum threshold alone on the original user-link graph to detect spammers. The threshold increases exponentially (base 2) from 1 to 16,384, and nodes are detected as spammers with the sum of their edge weights larger than or equal to the threshold. Figure 7(b) shows that applying this threshold alone could detect spammers with a false positive rate of 10.7% and a false negative rate of 3.6% when the threshold is set to 256. Although the performance of applying the threshold alone is not satisfactory, it does show that using this threshold can help improve the detection performance.

5.2 UNIK Evaluation



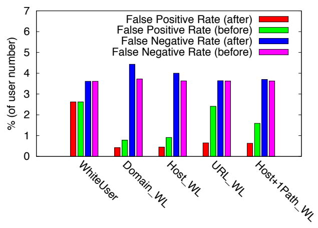
(a) Comparing different types of whitelist, varying the edge-weight-sum threshold



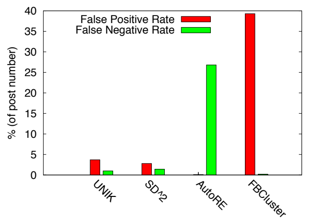
(b) CDF of edge-weight-sum in user-link graph

Figure 8: Evaluation of Whitelist and Edge-weight-sum

We now evaluate the effectiveness of UNIK with the help of the whitelist and the edge-weight-sum threshold together. Figure 8(a) shows the spammer detection results. The Host+1Path based whitelist detects spammers with a false positive rate of 0.6% and a false negative rate of 3.7% when the edge-weight-sum threshold is 256. The host-based whitelist has a false positive rate of 0.5% and a false negative rate of 4.0% with the same threshold. In contrast, WhiteUser and the all-host-based whitelist show worse performance. For WhiteUser, only non-spammers identified based on the social graph are removed from the user-link graph, and no edges are further removed for other non-spammers. As a result, applying the edge-weight-sum threshold can only help improve the performance to a limited extent.



(a) Applying the edge-weight-sum threshold (256) before or after applying the whitelist



(b) Comparing UNIK with SD2, AutoRE and FBCluster (in post number)

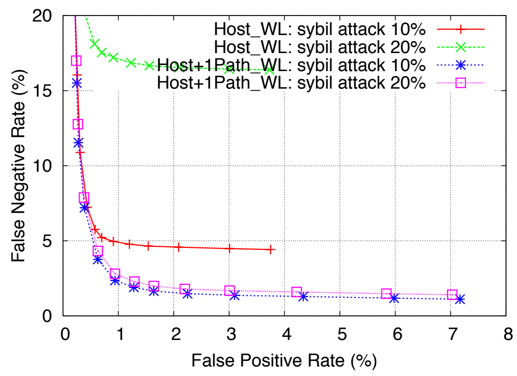
Figure 9: Evaluation of UNIK

Figure 8(b) shows the distribution of the sum of edge weights for spammers and non-spammers in the user-link graph using the host-based whitelist. Most spammers have a sum of more than 64, while more than 70% of non-spammers have a sum less than 64 before the edge trimming. After the edges are trimmed by the whitelist, more than 80% of non-spammers have a sum less than 64. This means the whitelist could help further differentiate spammers with non-spammers when applying the edge-weight-sum threshold. Note that although the threshold cutoff still marks 20% of non-spammers in the user-link graph as spammers, the small number of non-spammers remaining in the graph results in a small number of false positives in the spam detection. Figure 9(a) shows that if we apply the edge-weight-sum threshold earlier before applying the whitelist, the performance is worse than applying the threshold after applying the whitelist. This further validates that trimming edges by the whitelist in advance is helpful to applying the edge-weight-sum threshold in detecting spammers. On the other hand, WhiteUser does not trim any edge, so it is indifferent to applying the threshold earlier or later.

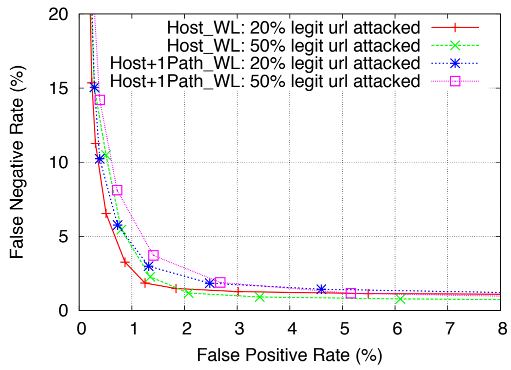
5.3 Comparisons with Other Schemes

Our UNIK scheme can detect spammers with a false positive rate of 0.6% and a false negative rate of 3.7%. In terms of the spam post being detected, the false positive rate is 3.7% and the false negative rate is 1.0% as shown in Figure 9(b). This suggests that UNIK is able to achieve the same level of detection performance as SD2. Figure 9(b) also shows that FBCluster [6] has high false positives. When we apply the suggested (5, 1.5 hrs) thresholds, the false positive rate is 39.3% while the false negative rate is 0.2%. We also have evaluated the AutoRE algorithm used in [19] by replacing the AS with the user account, which however has a 26.8% false negative rate when applying the suggested 20-AS and 5-day thresholds. Section 2 shows that tuning the threshold for FBCluster and AutoRE still cannot improve their performance.

5.4 Social Network Sybil Attack



(a) Evaluation of sybil attacks on spammer detection



(b) Evaluation of legitimate URL inclusion attacks

Figure 10: Evaluation of UNIK under attacks

Although UNIK works well in the evaluations with our dataset, its effectiveness is still subject to other possible enhanced attacks launched by spammers. To investigate the robustness of UNIK under attacks, we first evaluate UNIK performance by launching sybil attacks to the social graph of our dataset. To do so, we randomly select 10% of users in the social graph, and add the same number of spammers connecting to these users, forming a scale free social network among them. Then we double the number of spammers connected to the social network.

Figure 10(a) shows the impact of this sybil attack on UNIK spammer detection performance. We observe that the host-based whitelist is subject to sybil attacks: the false negative rate increases substantially when the sybil attack increases. This is because an increased level of sybil attacks increases errors in generating the host-based whitelist, trimming corresponding spam URL edges, and resulting in false negatives. However, if the whitelist is based on `Host+1Path`, then the errors are limited to URLs strictly matching the prefixes, which effectively limits the false negatives. In facing such attacks in the social graph, we should choose to limit the whitelist coverage for better performance. The false positive rate is 0.6% and the false negative rate is 4.3% for `Host+1Path` based whitelist with the edge-weight-sum threshold of 256, when the sybil attack intensity is 20% of social network users. This indicates that UNIK overcomes the limitation of SD2 when sybil attacks increase in the social network.

5.5 Legitimate URL Inclusion Attack

Similar to the sybil attack occurred in the social graph, spammers can include legitimate URL in the spam so that they can escape from being detected. Such inclusion increases the connectivity of the spammers and non-spammers in the user-link graph. For this kind of attacks, spammers need to crawl legitimate URLs of non-spammers in the user-link graph, which is often limited in crawling speed or scope.

We simulate legitimate URL inclusion attack to our dataset by assuming the spammers have made the effort crawling a major portion of total legitimate URLs in the user-link graph, and have inserted the crawled legitimate URLs into every spam posts. Since the total number of legitimate URLs is much larger than that of spam URLs, each spam post in this simulated attack only contains a minor fraction of spam URLs while the majority of URLs are legitimate.

Figure 10(b) shows the evaluation results of UNIK under such attacks. For a legitimate URL being attacked, it will not be included in the whitelist, as the majority of sharing users of that URL are mainly spammers after the attack. However, the host or `Host+1Path` prefix of the URL may still be included in the whitelist, because it is unlikely that other URLs with the prefix are all attacked. Even if the URL prefix is not included in the whitelist, the edges of attacked non-spammers will still be trimmed by the whitelist if not all URLs are attacked. As shown in Figure 10(b), the host-based whitelist shows slightly better performance due to its wider coverage which is harder to be defeated by the attack. The `Host+1Path` based whitelist still has a false positive rate of 2.7% and a false negative rate of 1.9% (applying the edge-weight-sum threshold of 256) even if 50% of legitimate URLs are attacked by spammers. In summary, UNIK works well under legitimate URL inclusion attacks for different kinds of whitelist.

5.6 Limitations of UNIK

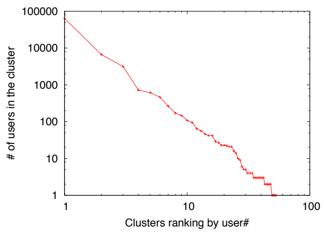
We have shown the promising performance of UNIK evaluated with our social blog dataset. And we also have shown the stable performance of UNIK facing difference scales of spam attacks. However, because UNIK is based on applying sybil defense scheme to detect spammers, UNIK has limitations in facing some types of spam attacks. For example, we have seen reports from [6] and [7] that a high percentage of spammers are using compromised user accounts to post spam in private social networks. The reason is that in private social networks, users primarily interact with a small list of other users, so spammers have to compromise normal users' accounts to widely promote their spam content. In this case, UNIK will have trouble to detect such compromised accounts as spammers since they are an integral part of the social graph. Therefore, UNIK is more suitable for fighting spam in an open social network such as groups, forums, or blogs, where the spamming activities are more intensive.

UNIK also needs to address the issue of URL shortening that is widely used in Twitter-like social networks. In facing shortened URLs, UNIK may need to fetch the final destination redirected by such a URL, so that the user-link graph can correctly represent the link sharing activities of different users. This increases the complexity of the system implementation of UNIK by introducing the cost of resolving URL redirects. In practice, we have seen systems incorporated such mechanisms in fighting social spam with reasonable cost [17].

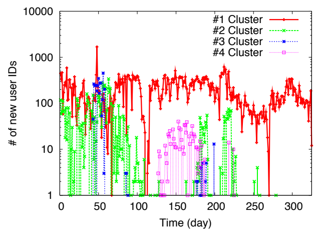
6. SPAM CLUSTER ANALYSIS

Based on the UNIK scheme presented in the last section, we are able to group spammers into different clusters in the user-link graph in our dataset. Each of these spammer clusters represents a group of spammer accounts that are interconnected with shared URLs in their posts, corresponding to different spam campaigns. Studying the spammer clusters can enable us to understand the patterns of different spam campaigns, and can also help develop spam signatures in fighting against future spam.

We first plot the number of spammer cluster sizes in Figure 11(a). Interestingly, the cluster size distribution follows power law, indicating that spam campaigns also have a natural distribution on their sizes. We are interested in the largest clusters, so we pick the top-4 clusters to study their characteristics. The top-4 clusters have 63699, 6634, 3159, and 724 spammer IDs, respectively. Figure 11(b) shows the number of new accounts created over time for each of the top-4 clusters. We observe that in #1 cluster new accounts were continuously created over the trace duration, while new accounts were created at different periods in other clusters. This clearly indicates that different spammer clusters have different activities patterns. There exists some correlation between the #1 cluster and the #3 cluster as their ID creation activities are both bursty around day 50.



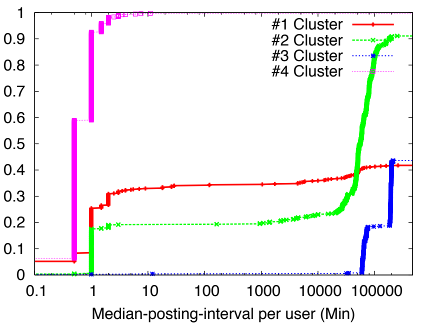
(a) Cluster size distribution



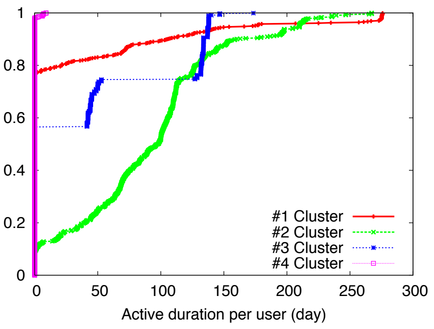
(b) Top-4 clusters: new IDs created daily

Figure 11: Spammer cluster size and activity over time

We also study the user activities of each spammer ID in the top-4 clusters. Figure 12(a) shows the median of posting interval for each user in the top-4 clusters. The #4 cluster demonstrates the shortest interval distribution: more than 90% IDs have a median of interval within 1 minute. On the contrary, the #3 cluster shows the longest interval distribution: most IDs have a median of interval larger than 40 days. Figure 12(b) shows the active duration of each ID in the top-4 spammer clusters. The #4 cluster shows a very short active duration while the #2 cluster shows a median duration of 100-day per user ID. The active duration distribution of #3 cluster shows three discretely stages including 0, 50, 137 days. The significant differences between different clusters imply that the behavior of spammer ID varies substantially. Therefore it is highly difficult to capture the spammer behavior with only a handful patterns.



(a) Top-4 clusters: median posting interval per user



(b) Top-4 clusters: active duration per user

Figure 12: Spammer cluster posting activity

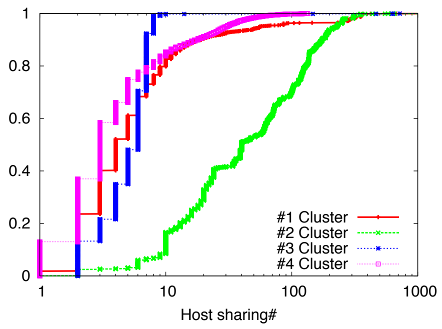


Figure 13: Top-4 clusters: host sharing intensity

Figure 13 shows the sharing intensity of each host in the top-4 clusters. The sharing intensity of a host is defined as the number of users that ever posted link(s) pointing to the host, which captures the intensity of advertising activities of a spam host. The # 2 cluster shows the strongest sharing intensity of the advertised spam hosts, while the other clusters show similar intensity distributions. This is correlated with the longest active duration distribution of # 2 cluster as shown in Figure 12(b). This finding implies that the host sharing intensity is increased with the increase of spamming activity by spammers over time.

7. OTHER RELATED WORK

Spam in online social networks has been actively studied through measurement-based analysis recently. For example, Thomas et al. [17] proposed a real-time system Monarch to detect spam based on URL-related features, and they found Twitter spam campaigns are long lasting than email spam campaigns. Grier et al. [7] analyzed spam in Twitter with public blacklists, and showed that 8% of all URLs are on popular blacklists, although the blacklists are too slow in preventing the damage of harmful URLs. Gao et al. [5] proposed social network features such as social degree for online spam filtering based on Facebook and Twitter datasets.

Several other approaches have been proposed to detect spam in emails with spam templates, network-level features, shared IP addresses, or email target domains. Qian et al. [14] proposed an unsupervised learning based email spam filter since spam from the same campaign often contains unchanged textual snippets by using the same template. Hao et al. [8] studied using network-level features to distinguish email spammers from non-spammers as the first level defense since it is difficult to maintain IP-based blacklists. BotGraph [21] evaluated a graph-based approach with the MapReduce model to cluster millions of bot-users observing that bot-users share IP addresses to log in or send emails. SpamTracker [15] tried to catch email spammers earlier than traditional IP-based blacklists do by using the target domains that a IP address sends emails to as a behavioral feature to cluster IP addresses.

8. CONCLUSION

Albeit a number of spam detection schemes have been designed, spam in online social networks increases significantly in recent years. In this study, we first analyze the limitations of existing representative schemes, and then design a sybil defense based spam detection scheme, SD2. SD2 is an unsupervised scheme, which outperforms existing unsupervised schemes significantly with the help of the social network graph. But it suffers from escalating sybil attacks to the social network. Thus, we further propose UNIK that is highly robust to an increased level of spam attacks. UNIK differs from existing schemes in that it detects non-spam URL patterns from the social network instead of spam URLs directly, because the non-spammer patterns are relatively more stable than spammers. UNIK demonstrates its performance with a 10-month social network dataset. Based on UNIK detection, we have also analyzed spammer clusters in the dataset, and find distinct spamming patterns of different spam campaigns, indicating the volatility of spamming patterns.

9. ACKNOWLEDGMENTS

We thank the anonymous reviewers for their constructive comments. The work was supported in part by the National Science Foundation under grants CCF-0913050, CNS-1162165, CNS-0746649 and CNS-1117300.

10. REFERENCES

[1] http://nakedsecurity.sophos.com/2011/01/19/sophos-security-threat-report-2011-social-networking/.

[2] Barracuda labs 2010 annual security report. http://www.barracudalabs.com/research%5Fresources.html.

[3] F. Benevenuto, T. Rodrigues, V. Almeida, J. Almeida, and M. Gonçalves. Detecting spammers and content promoters in online video social networks. In *Proc. of SIGIR*, 2009.

[4] G. Danezis and P. Mittal. SybilInfer: Detecting sybil nodes using social networks. In *Proc. of NDSS*, 2009.

[5] H. Gao, Y. Chen, K. Lee, D. Palsetia, and A. Choudhary. Towards online spam filtering in social networks. In *Proc. of NDSS*, 2012.

[6] H. Gao, J. Hu, C. Wilson, Z. Li, Y. Chen, and B. Y. Zhao. Detecting and characterizing social spam campaigns. In *Proc. of IMC*, 2010.

[7] C. Grier, K. Thomas, V. Paxson, and M. Zhang. @spam: The underground on 140 characters or less. In *Proc. of ACM Conference on Computer and Communications Security (CCS)*, 2010.

[8] S. Hao, N. Syed, N. Feamster, A. Gray, and S. Krasser. Detecting spammers with SNARE: Spatio-temporal network-level automatic reputation engine. In *Proc. of USENIX Security Symposium*, 2009.

[9] K. Lee, J. Caverlee, and S. Webb. Uncovering social spammers: Social honeypots + machine learning. In *Proc. of SIGIR*, July 2010.

[10] J. Leskovec, K. J. Lang, A. Dasgupta, and M. W. Mahoney. Statistical properties of community structure in large social and information networks. In *Proc. of WWW*, 2008.

[11] B. Markines, C. Cattuto, and F. Menczer. Social spam detection. In *Proc. of AIRWeb*, 2009.

[12] A. Mohaisen, A. Yun, and Y. Kim. Measuring the mixing time of social graphs. In *Proc. of IMC*, 2010.

[13] A. Pathak, F. Qian, Y. C. Hu, Z. M. Mao, and S. Ranjan. Botnet spam campaigns can be long lasting: evidence, implications, and analysis. In *Proc. of SIGMETRICS*, 2009.

[14] F. Qian, A. Pathak, Y. C. Hu, Z. M. Mao, and Y. Xie. A case for unsupervised-learning-based spam filtering. In *Proc. of SIGMETRICS*, 2010.

[15] A. Ramachandran, N. Feamster, and S. Vempala. Filtering spam with behavioral blacklisting. In *Proc. of CCS*, 2007.

[16] E. Tan, L. Guo, S. Chen, X. Zhang, and Y. E. Zhao. BARS: Spam detection in user generated content. In *Proc. of ICDCS*, 2012.

[17] K. Thomas, C. Grier, J. Ma, V. Paxson, and D. Song. Design and evaluation of a real-time url spam filtering service. In *Proc. of IEEE Symposium on Security and Privacy*, 2011.

[18] B. Viswanath, A. Post, K. P. Gummadi, and A. Mislove. An analysis of social network-based sybil defenses. In *Proc. of SIGCOMM*, 2010.

[19] Y. Xie, F. Yu, K. Achan, R. Panigrahy, G. Hulten, and I. Osipkov. Spamming botnet: Signatures and characteristics. In *Proc. of SIGCOMM*, 2008.

[20] H. Yu, P. Gibbons, M. Kaminsky, and F. Xiao. SybilLimit: A near-optimal social network defense against sybil attacks. In *Proc. of IEEE Security and Privacy*, 2008.

[21] Y. Zhao, Y. Xie, F. Yu, Q. Ke, Y. Yu, Y. Chen, and E. Gillum. BotGraph: Large scale spamming botnet detection. In *Proc. of NSDI*, 2009.