SocialWatch: Detection of Online Service Abuse via Large-Scale Social Graphs

Junxian Huang[1] Yinglian Xie[2] Fang Yu[2] Qifa Ke[2]
Martín Abadi[2]* Eliot Gillum[3] Z. Morley Mao[1]
[1]University of Michigan [2]Microsoft Research Silicon Valley [3]Microsoft Corporation

ABSTRACT

In this paper, we present a framework, SocialWatch, to detect attacker-created accounts and hijacked accounts for online services at a large scale. SocialWatch explores a set of social graph properties that effectively model the overall social activity and connectivity patterns of online users, including degree, PageRank, and social affinity features. These features are hard to mimic and robust to attacker counter strategies. We evaluate SocialWatch using a large, real dataset with more than 682 million users and over 5.75 billion directional relationships. SocialWatch successfully detects 56.85 million attacker-created accounts with a low false detection rate of 0.75% and a low false negative rate of 0.61%. In addition, SocialWatch detects 1.95 million hijacked accounts—among which 1.23 million were not detected previously—with a low false detection rate of 2%. Our work demonstrates the practicality and effectiveness of using large social graphs with billions of edges to detect real attacks.

Categories and Subject Descriptors

G.2.2 [**Graph Theory**]: Graph algorithms; K.6.5 [**Security and Protection**]: Authentication

Keywords

SocialWatch, social graph, spam, PageRank, security

1. INTRODUCTION

Virtually all large-scale online services today (*e.g.,* Hotmail, Facebook) have become popular platforms for attackers to conduct a wide range of nefarious activities. Although the precise methods or scales of these attacks may differ, the common thread among them is the requirement of a large number of malicious user accounts. These accounts can be either created anew or obtained via compromising real user accounts. They serve as the channels to propagate spam, malware, social scam, or other malicious contents.

Many existing security mechanisms to deal with malicious accounts have extensively studied the attack characteristics [1, 2, 3, 4]. However, as attacks evolve over time, one has to constantly adapt features for detection in the arms race. Thus, two important questions naturally arise: Are there robust features for distinguishing malicious accounts from legitimate ones? If so, are they practical to compute?

In this paper, we explore the use of *social graph features* for detecting both attacker-created accounts and compromised accounts at a large scale. The motivation is that a social graph describes not just the behaviors of individual user accounts, but also the interactions among them and the structures of their relationships. Although attackers can easily adjust or adapt the behaviors of the accounts that they control, they cannot change the behaviors of other legitimate accounts. For example, most legitimate users do not respond to spam emails and attackers cannot force them to reply. Therefore, attackers will have limited effect on the local graph features of legitimate users. Furthermore, it is fundamentally difficult for them to change the overall communication patterns involving other legitimate users. The larger the legitimate user population is, the harder it is for attackers to influence the global social graph structures.

We design and implement a framework called SocialWatch to detect attacker-controlled accounts using social graphs. Unlike previous work that focuses on a specific feature, SocialWatch explores a set of graph properties. Some of them are local graph features based on degree and PageRank. However, we show that directly applying degree and PageRank to our context is not effective as legitimate users have diversified behaviors and attackers can be stealthy. Therefore, we modify these well-known features and use them in combination.

In addition, to address the challenges of detecting hijacked accounts with mixed behaviors of normal and malicious activities, we propose a set of *social affinity features* that not only describe individual account behaviors, but also the behaviors of its contacts and the community structures. We leverage these fine-grained social features and use a Bayesian decision framework for detection. We show that, despite the presence of noise and missing information in the data set, the selected social affinity features can still effectively differentiate hijacked accounts from normal ones. In summary, SocialWatch puts attack detection in a large social context and hence is robust to attacker counter strategies.

To make our solution practical, all the graph features that we select can be efficiently computed using the cloud computing techniques today. We implement SocialWatch on a cluster of 240 machines using Dryad/DryadLINQ [5] and evaluate it using a large dataset with more than 682 million users and 5.75 billion directional relationships. We demonstrate the practicality and effective-

*Martín Abadi is also affiliated with the University of California, Santa Cruz.

ASIA CCS'13, May 8–10, 2013, Hangzhou, China.

ness of our approach by detecting real, large-scale attacks from the dataset. We summarize our key results as follows:

- Using algorithms that combine degree and PageRank features together, SocialWatch detects 56.85 million attacker-created accounts, with a 0.75% false detection rate and a 0.61% false negative rate.
- Using the fine-grained social affinity features, at a false detection rate of 2%, SocialWatch identifies 1.95 million hijacked accounts, among which 1.23 million were not detected previously.

The rest of the paper is organized as follows. We discuss related work in §2. We present our problem formulation in §3 and describe the detection methodology in §4. We evaluate the evaluation results in §5, before concluding in §6.

2. RELATED WORK

Detecting malicious accounts for online services has been an important problem in recent years. Existing studies have proposed to identify correlated abnormal behaviors for detecting such attacks at a large scale [4, 1]. Although these defense systems are demonstrated to work effectively for existing attacks, attackers may evade by modifying the behaviors of multiple malicious accounts simultaneously to make them look different. The goal of our study is to design a more robust social framework by looking at not only individual user behaviors, but also the interactions among all users.

Sybil attacks and defense in social networks have been extensively studied [6, 7, 8, 9]. Most existing Sybil-defense solutions are designed for distributed environments. Further, recent studies show that some of the key assumptions required by Sybil solutions may not hold on real social graphs [10]. In contrast, our approach takes a centralized view of the entire social graph, allowing us to explore more efficient and practical solutions. As we focus on attack detection, our work is also different from the recent work that focuses on identifying legitimate user populations early using social graphs [11].

With Online Social Networks (OSN) growing rapidly, there exist extensive studies for understanding their graph properties [12, 13, 14]. Their findings are important and complementary to our work, where we focus on leveraging their findings and various graph properties for improving security.

There also exist previous studies that explore social graph features for attack detection [15, 16, 17, 18]. The early success of these proposals motivate us to further explore richer graph properties and use them in combination. In addition, we also explore more fine-grained graph features that capture community structures and user social distances for detecting hijacked accounts.

3. PROBLEM FORMULATION

We leverage social graphs to detect both attacker-created accounts and hijacked accounts. Although SocialWatch is a general solution that is applicable to different types of online services, in this paper, we use an email application to drive our presentation due to the use of our dataset.

3.1 Input Dataset

We have access to a large data set collected from Microsoft Hotmail. The dataset contains coarse-grained communication information of anonymized, sampled user accounts from October 2007 to April 2010. Based on whether the email address is registered with Hotmail, we classify accounts into *internal accounts* and *external accounts*. Each entry in the data set contains the communication history summary between a sampled internal user account and another user (either internal or external), including the anonymized email addresses of both users and the number of emails that the two users have mutually sent to each other.

Note that the sampling method is user-based. In other words, if an internal user is sampled, the dataset records all other users that this sampled user has ever communicated with during the data collection period. Thus for each sampled account, we have a complete view of its communication patterns to construct graphs. For other internal accounts and external accounts, we have only a partial view of their communications, derived from the records of the set of sampled internal accounts that they have communicated with.

In total, our dataset contains 5.745 billion directional records involving more than 682 million unique accounts, among which 269.5 million are internal accounts.

3.2 Graph Construction

Given the input, SocialWatch builds two social graphs. On both graphs, a node v corresponds to a unique user, but the edges are defined differently.

The first graph is a *directed email-communication graph* $G_d = (V, E_d)$, which records the detailed email activities between users. Each directed edge $e(v_1, v_2) \in E_d$ represents that node v_1 has sent emails to v_2, and the weight w of this edge indicates the number of emails that v_1 has ever sent to v_2 during the data collection period.

The second graph is an *undirected friendship graph* $G_u = (V, E_u)$, which records only the mutual relationship between users. The edges in E_u are extracted from the friendship patterns of E_d, *i.e.,* if both $e(v_1, v_2) \in E_d$ and $e(v_2, v_1) \in E_d$, and the weight of the two edges are at least 2, we define an undirected edge e' that connects v_1 and v_2 in G_u. By requiring mutual email exchanges on the undirected graph, SocialWatch attempts to exclude connections from malicious accounts to legitimate accounts, as legitimate users usually do not reply to spam emails. The edge weight represents the strength of the connection between two accounts. By requiring an edge to have a weight of at least 2, SocialWatch prunes weak connections due to occasional or accidental email exchanges (*e.g.,* accidentally reply to a malicious account). The constructed G_d has 682 million nodes with degree ≥ 1 and 5.745 billion edges, and G_u includes 255 million nodes and 436.9 million edges.

3.3 Problem Statements and Challenges

Given both the directed email-communication graph and the undirected friendship graph, our goal is to develop methods to detect the set of nodes tat correspond to either attacker-created accounts or hijacked accounts. In [19], we investigate in detail the adversary behaviors of these two types of accounts on the social graphs that we created. We aim to maximize detection coverage while minimizing the false detection rates, *i.e.,* the percentage of legitimate users that are mistakenly marked as malicious over the total number of users that are marked as malicious by our methods. Selecting robust and effective graph properties for malicious account detection is challenging, especially under our practical settings:

- **Incomplete graph**. Any large graphs constructed in practice have only a partial view based on the specific set of accounts (either due to sampling or the limited view from one provider). Therefore, the graph-based detection algorithms must be robust to missing data.
- **Large-scale graph**. As our graph is huge with billions of edges, the selected graph properties need to be computed efficiently on large graphs.
- **Legitimate accounts have diverse behaviors**. Given our graphs have hundreds of million users, the behaviors of legitimate accounts can be highly diversified. For example, some

legitimate accounts, such as mailing-lists, may have thousands of contacts. On the other hand, attackers could choose to send spam to only tens of victims from each malicious account to evade detection. Our method needs to distinguish both aggressive and stealthy malicious accounts from the remaining ones.

- **Hijacked accounts have mixed behaviors**. Traditional graph properties such as PageRank and degrees may not able to provide strong signals. Hence, more fine-grained graph properties are required to detect the subtle differences between hijacked accounts and the remaining legitimate ones.

Due to the above challenges, there is no single graph property that can distinguish malicious accounts from legitimate ones. Therefore, SocialWatch resorts to a set of complementary graph properties. For each account, we are interested in not only its own behaviors, but also the behaviors of its contacts, or even all the other accounts to some extent. Putting the detection in a large social context is the key for the social graph based detection.

4. DETECTION METHODOLOGY

In this section, we present our detection methodology for attacker-created accounts and hijacked accounts.

4.1 Detecting Attacker-created Accounts

There are two most prominent and easy-to-compute graph properties that are widely used for detection: node degree and PageRank. Degree is a *local* graph feature that captures the aggressiveness of an account. PageRank is a *global* graph feature that calculates the weights of the nodes on the overall graph. We next discuss how we modify these two properties in the social context for detection.

4.1.1 Defense Using Degree

The straightforward degree-based method of disallowing any account sending emails to more than N recipients may affect the usability of active legitimate users. Therefore, we introduce the *response rate* of an account, *i.e.,* the ratio between the number of replied recipients to the total number of recipients. Note that we use the number of recipients rather than the number of emails because a friend would typically reply to *at least one* emails sent over the entire history, but may not respond to *all* emails. Therefore, the number of recipients is a better metric to quantify friendship than using the number of emails. For spam emails, most users do not reply to *any* of them, so spammers typically have low response rates.

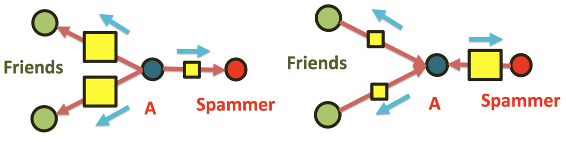
4.1.2 Defense Using PageRank

PageRank is widely used for ranking Web pages and recently has been applied to social graphs to detect spammers [17]. Each node is assigned a uniform reputation score initially. In each iteration of the PageRank computation, each node propagates its reputation to neighbors. After the ith iteration, the node A's new reputation score $R_{A,i+1}$ is given as

$$R_{A,i+1} = 1 - d + d \sum_{\{X:e_{XA} \in E\}} \frac{R_{X,i}}{outdegree(X)} \qquad (1)$$

where d is the damping factor usually set to 0.85 [20]. $R_{X,i}$ is the reputation score of node X after the previous iteration, and $\{X : e_{XA} \in E\}$ is the set of nodes in the graph that have directed edges pointing to A.

One method is to directly apply PageRank to the directed G_d, where the PageRank scores propagate along directed edges. Intuitively, the PageRank of a node models the *goodness* of the user, and we call it the *goodness score*. However, even with a response rate as low as 1%, an aggressive spammer who sent thousands of emails may still receive emails from legitimate accounts (*e.g.,* accidental replies) and get a high goodness score. On the other hand, inactive, legitimate accounts who do not receive many emails may have a low goodness score.



(a) goodness score (b) badness score

Figure 1: Goodness and badness score propagation. The box size is in proportion to the edge weight[1].

To mitigate this problem, we introduce two modifications to this approach. The first is that we adjust edge weights based on email exchange patterns. We use the number of emails exchanged between two accounts to infer this relation. Assume that an account A has n contacts (C_i, $1 \leq i \leq n$), and A sends s_i emails to C_i, while receiving r_i emails from C_i. The percentage ω_i of the goodness score that A should propagate to C_i is defined by

$$\omega_i = \left(\frac{s_i+1}{r_i+1}\right) \Big/ \sum_{k=1}^{n} \left(\frac{s_k+1}{r_k+1}\right) \qquad (2)$$

Here, we add 1 to s_i and r_i to avoid the divided-by-zero problem. As in Figure 1(a), A would propagate more fraction of her score to friends who have "balanced" mutual email exchanges with her, while less to the spammers who send multiple emails to A with few or zero replies.

The second modification we adopt is the reverse PageRank [21], calculated by running the PageRank algorithm on the social graph obtained by reversing the directions of all edges as illustrated in Figure 1. We name the score of a node obtained in this way a *badness* score. The intuition behind the badness score is that bad accounts normally send many emails, so if we propagate badness scores from recipients to senders, bad accounts would receive high badness scores. Similarly, the edge weight is adjusted based on email exchange patterns so that a node would propagate more of her badness score to spammers and less to her friends.

We compute the ratio of goodness to badness scores by PageRank computation and use this ratio to detect spammers. We further combine the modified PageRank and degree-based approaches to achieve better detection results using the following steps:

- Identify aggressive spamming accounts with high out degrees ($\geq N'$) and low response rates ($\leq \theta$).

- Identify less aggressive spamming accounts using the badness-goodness PageRank ratio ($\geq \alpha$).

The settings of N', θ, α are based on a small set of known good and bad accounts [19].

4.2 Methods for Hijacked Account Detection

The communication patterns of hijacked accounts are different from attacker-created ones, *e.g.,* hijacked accounts will continue to receive emails from their contacts, while attacker-created accounts often do not receive emails from legitimate users. The degree and PageRank-based approaches in §4.1 thus may not catch hijacked accounts. Instead, we explore fine-grained graph properties and use two social-affinity features to detect hijacked accounts.

[1] Circles in the graph represent user nodes. Boxes denote the propagation of goodness/badness score and the box size is proportional to the edge weight.

4.2.1 *Social-Affinity Features*

We introduce social-affinity features defined for each user v, aiming to capture the subtle differences between hijacked accounts and legitimate accounts.

Recipient connectivity r: A legitimate user usually belongs to one or a few communities in the real world based on geographic locations, working connections, or interest, and so are her recipients. For example, we have observed strong evidence of geographic locality in communication patterns—80.7% of communication edges are between users within a same country.

In contrast, spam emails are often sent to users who are not socially connected. It is difficult for attackers to identify community patterns and mimic normal user behaviors. Thus, how well the email recipients of a given user v are socially connected can be defined as a social affinity feature for hijacked account detection.

The recipient connectivity for a user v is defined based on how well $R(v)$, the recipients of v, are connected in the undirected friendship graph G_u. SocialWatch computes the fraction of such connected recipients among $R(v)$. For each account v, $R(v)$ forms a subgraph $G_u(v)$. Note that $G_u(v)$ does not include v. SocialWatch then identifies all the connected components $C_1 \cdots C_k$ with size of at least two on the subgraph $G_u(v)$. Let $R_i(v)$ denote the set of recipients in C_i ($1 \leq i \leq k$). Then the recipient connectivity feature $r(v)$, the fraction of socially connected recipients, is defined as:

$$r(v) = \frac{\sum_{i=1}^{k} |R_i(v)|}{|R(v)|} \tag{3}$$

Social distance l: In practice, the email communication graph is only a sub-graph of the real-world communication graph. For example, we may not be able to observe the connections among accounts whose communication records are not included in our dataset (e.g., accounts external to the service that we are protecting). Such incomplete graph may lead to incorrect measurements in the recipient connectivity feature $r(v)$, because two external recipients of v that are socially connected in the real world are now disconnected (not belonging to the same connected component) in our observed graph G_u.

We use a social distance $d(v_1, v_2)$ on the friendship graph G_u to describe how close two users v_1 and v_2 are on the social graph. It is defined by the *distance of the shortest path* between v_1 and v_2 on G'_u, where G'_u is defined by removing v (and edges connected with v) from G_u.

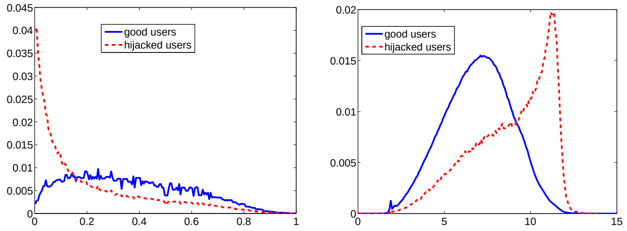
We define the second social feature $l(v)$ for user v as the mean of all pairwise social distances between any two users in $R(v)$:

$$l(v) = \frac{1}{N} \sum_{v_i, v_j \in R(v), i<j} d(v_i, v_j) \tag{4}$$

where $N = |R(v)| * |R(v) - 1|/2$ is the number of all possible user pairs.

In the case of missing observations mentioned above, the social distance defined by the shortest-path distance is an upper-bound approximation of the real social distance on the complete graph. $l(v)$ is robust to missing data and complementary to the recipient-connectivity feature $r(v)$.

The social affinity features $r(v)$ and $l(v)$ exploit the structure of the subgraph defined by the email recipients of the user v. We use $r(v)$ and $l(v)$, combined with node out-degree (denoted as $D(v)$), for hijacked account detection. Previous work has leveraged clustering coefficients for similar purposes [16]. We do not use clustering coefficients because they only account for direct connections between pair-wise accounts and are thus sensitive to missing data.



(a) $r(v)$ (b) $l(v)$

Figure 2: Social affinity features for detecting hijacked accounts.

4.2.2 *Computing Social Affinity Features*

The recipient connectivity feature $r(v)$ is relatively easy to compute in parallel for all accounts. However, the social-distance feature $l(v)$ requires computing shortest-path distances on the large graph G_u for all user pairs in $R(v)$, which is computational expensive and hard to parallelize. SocialWatch adopts a sketch-based approach to compute an approximate shortest-path distance between a pair of nodes using sampled landmark nodes [22]. Using this algorithm, the estimated path length is slightly larger than the actual value with provable bounds. We find that the approximation works well in practice for our purpose of detecting hijacked accounts.

Figure 2 shows the distributions of the two social-affinity features for a set of known hijacked accounts and another set of sampled legitimate accounts, both labeled by the email provider. As we can see, for each feature, the distribution of hijacked accounts is significantly different from that of legitimate accounts. The distribution of $r(v)$, the recipient-connectivity feature, is relatively even between 0 and 1 for legitimate users, but is heavily peaked at zero for hijacked users, which means that the email recipients of a hijacked user are mostly disconnected. The average pairwise shortest-path distance $l(v)$ follows a normal distribution with a mean of about 7 for legitimate accounts, but is peaked at about 12 for hijacked accounts.

These two social-affinity features are complementary to each other as they capture different social graph properties of a user. In the following we present two scenarios for using these features together to detect hijacked accounts. The first scenario is when we do not have pre-labeled hijacked accounts, i.e., we want to detect hijacked accounts based on the properties of legitimate accounts. The second scenario is a refined approach when we are provided with a set of known hijacked accounts.

4.2.3 *Detection Without Known Hijacked Accounts*

For legitimate accounts, the distributions of their graph features are usually stable over time and can be estimated accurately. Without known hijacked accounts, we can use one-tailed hypothesis testing [23] to detect hijacked accounts. The null hypothesis is:

$$H_0 : \omega = W_0 \tag{5}$$

where ω denotes the unknown account label, and W_0 denotes the normal account label. Given the observed features $\mathbf{x}^0$ of a user v (in our case $\mathbf{x}^0 = \{D(v), r(v), l(v)\}$), and assuming the null hypothesis is true, we can estimate the one-tailed critical p-value (the statistical significance level):

$$P_0 = P(\mathbf{x} > \mathbf{x}^0 | \omega = W_0) = \int p(\mathbf{x} > \mathbf{x}^0 | \omega = W_0) d\mathbf{x} \tag{6}$$

For simplicity, we assume that the social features are independent. Equation 6 becomes:

$$P_0 = \prod_{i=1}^{3} \int p(x_i > x_i^0 | \omega = W_0) dx_i \tag{7}$$

Symbol	N	N'	θ	α
Description	Inactive limit	Aggressive limit	Response rate	Badness/ goodness ratio
Value	5	500	5%	4.5

Table 1: Selected values for detection thresholds.

If P_0 is less than a predefined statistical significance level t, then we reject the hypothesis. The significance level t is a parameter to control the potential false detection rate of such detection. Specifically, given t, SocialWatch computes a threshold along each feature dimension based on data. We classify an account as a hijacked account if one of its feature values violates the computed threshold.

4.2.4 Detection With Known Hijacked Accounts

In practice, a service provider can often identify a subset of hijacked accounts based on user reports, or manual examination, *etc.* Although the distribution of hijacked accounts is not as stable as normal accounts, we can still leverage the knowledge of known hijacked accounts to detect remaining ones. SocialWatch uses a Bayesian decision framework to detect additional hijacked accounts using the known ones as training data.

Given the learned probability distributions of both normal and hijacked accounts, the Bayesian decision theory [24] gives the following decision boundary:

$$\frac{p(\mathbf{x}|W_1)}{p(\mathbf{x}|W_0)} > \frac{c_0}{c_1}\frac{P(W_0)}{P(W_1)} \tag{8}$$

where W_1 denotes the class of hijacked accounts. Here c_0 represents the cost (*i.e.,* penalty) of labeling a normal account as a hijacked account, and c_1 the cost of labeling a hijacked account as a normal account. Assuming that features are independent, the decision ratio can be directly computed from the learned probability density functions shown in Figure 2(a)-(c). If the above inequality holds, $\mathbf{x}$ is classified as hijacked, otherwise it is classified as normal. $P(W_0)$ is the prior of a given account to be a normal account and $P(W_1)$ the prior of being a hijacked accounts. Both $P(W_0)$ and $P(W_1)$ can be estimated by the fraction of normal and hijacked accounts in the training set. The only parameter to set is the ratio of $\frac{c_0}{c_1}$, which is independent of the observed data, and is intuitive thus easier to understand and control by the service providers.

5. EVALUATION

We implement SocialWatch using Dryad/DryadLINQ [5] and it processes data in parallel on a 240-machine cluster. In this section, we evaluate SocialWatch on the large graphs that we build (see Section 3.2). We describe how we label the ground-truth of data for validation in the longer version of this paper [19].

5.1 Degree and PageRank Based Detection

After filtering inactive users, we first detect all the active users using degree and PageRank based method as described in §4.1.

We pick a small set of labeled accounts as training data to derive the threshold parameters (listed in Table 1) and we exclude them from our evaluation.

Using the degree-based method alone, SocialWatch detects a vast majority of the attacker-created accounts with a low false detection rate (1.06%) and a low false negative rate (1.19%). Indeed, legitimate and malicious accounts exhibit very different patterns in their email response rates. In our data set, more than 95% of the bad accounts have 0 in-degrees on the directed graph, meaning they receive no replies. In contrast, the median response rate of good accounts is over 50% and only 5% of them receive no replies (most of them are actually inactive users with small out-degrees).

We further apply PageRank-based detection to increase the detection coverage and reduce the false detection rate. It takes about 30 iterations for the goodness and badness scores of all the nodes to converge ($\leq 1\%$ difference between two consecutive iterations).

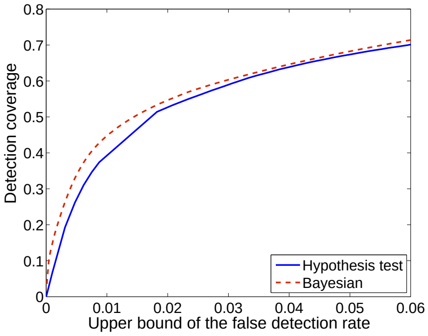


Figure 6: Detection coverage vs. the upper bound of false detection rate for hijacked users (the newly detected users are all regarded as false positives in this figure).

After obtaining the badness-goodness PageRank ratio, we need to select a cutoff threshold. To do so, we leverage the training data mentioned earlier. Figure 3 shows the CDF of their badness-goodness ratio. We find good and bad accounts in the training data have clear distinctions. Accordingly, we select $\alpha = 4.5$ as the threshold for detection on the entire graph. In total, among all 269.5 million internal accounts, we detect 56.85 million bad accounts by combining degree and PageRank in detection and reduce the false detection rate to 0.75%. Among the falsely detected accounts, we find some of them have a high badness score because they sent emails to a few very large mailing-list addresses (which typically have high badness scores). Thus we can potentially improve our detection further by removing large mailing lists from detection. Our false negative rate is as low as 0.61%. Among those false negative users, 38.88% are actually known hijacked accounts that we fail to detect using degree and PageRank.

The above results suggest that degree and PageRank can effectively detect attacker-created accounts accurately. For the set of detected accounts, service providers can apply more strict policies to further validate them (*e.g.,* giving more CAPTCHA or asking security questions) while throttling their attack behaviors substantially (*e.g.,* by limiting the email-sending rates aggressively).

5.2 Social Affinity Based Detection

After applying the combined degree and PageRank algorithm to the email graph to detect attacker-created accounts, we apply the social-affinity based method on the remaining users to detect hijacked accounts. In particular, we leverage the two detection methods, namely *hypothesis testing* and *Bayesian decision* introduced in §4.2.1. The hypothesis testing does not need known hijacked accounts, so we use the labeled good accounts to build normal account distributions and use the remaining users as testing data. For the Bayesian decision method, we use half labeled good accounts and half known hijacked accounts for training, and the remaining data for testing.

Figure 6 shows the detection coverage vs. false detection rate from hypothesis testing (solid curve) and Bayesian decision (dashed curve). We vary the false detection rate from 0 to 6%. The detection results are compared to the labels provided by the email provider. We observe that Bayesian decision performs slightly better. At a 6% false detection rate, hypothesis testing detects about 70% of the hijacked accounts, while Bayesian decision detects about 72%. At a false detection rate of 2%, using hypothesis testing, SocialWatch detects 53.3% hijacked accounts. One advantage of using the Bayesian approach is that the parameter has an intuitive meaning, and is easier to understand and control.

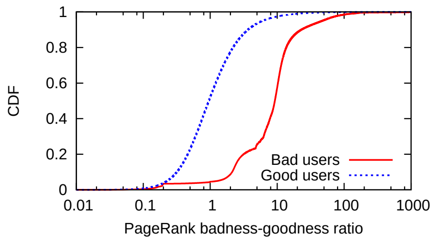


Figure 3: PageRank **badness-goodness ratio.**

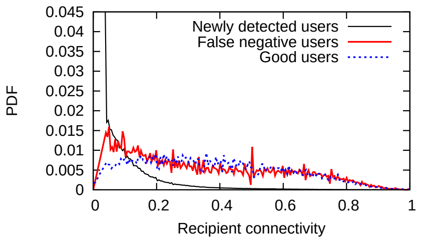


Figure 4: Recipient connectivity Eq 3[2].

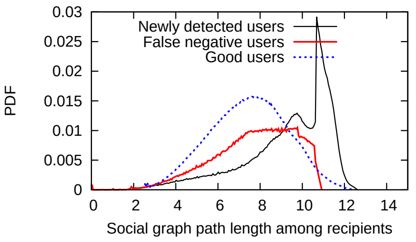


Figure 5: Social distance among recipients Eq 4.

Our method also flags 1.53 million additional accounts as being hijacked. These accounts are our new findings, which have not been reported by the existing system yet. Similarly, using Bayesian decision, SocialWatch detects 54.6% known hijacked accounts as well as 1.23 million additional ones.

For the newly detected accounts, the question is "are they indeed hijacked users that were not detected yet?" We study them using two approaches. First, we submitted these accounts to the email provider for manual verification based on sampling. They confirmed that the sampled accounts exhibit suspicious email activities. After a year, most of them were either reported by the users or captured by content-based detection. Second, we compare two social-affinity features of these accounts with the overall good account population (Figure 4, 5). These two features describe the community structures of the accounts. We observe that the newly detected accounts have clearly different feature distributions from the good account population, *i.e.,* with smaller ratios of recipient connectivity $r(v)$ and larger social distances $l(v)$. In fact, their distributions are similar to the set of known hijacked accounts. These accounts thus may already have started spamming activities at the time of our detection.

On the other hand, for the false negative accounts, i.e., the set of hijacked accounts that we fail to detect, we find that they have similar social feature distributions to the known good ones. These may be more stealthy accounts or the set of accounts not actively misbehaving yet, making it inherently challenging to detect them.

6. CONCLUSION

In this paper, we present an online service protection framework, SocialWatch, that uses social connectivity features to detect attacker-created accounts and hijacked accounts at a large scale. We explore a rich set of graph properties including degree and PageRank that are effective for detecting aggressive attack behaviors, as well as social affinity metrics that capture the subtle differences between the stealthy hijacked accounts and the remaining legitimate users. We show that SocialWatch can effectively detect tens of millions of malicious accounts in the wild. Since the social connectivity features that we rely on can be derived from a wide class of applications, we believe that SocialWatch is general and promising for protecting both the providers and the online users of many services.

7. REFERENCES

[1] Y. Xie, F. Yu, K. Achan, R. Panigrahy, G. Hulten, and I. Osipkov, "Spamming Botnets: Signatures and Characteristics," in *SIGCOMM*, 2008.

[2] S. Hao, N. A. Syed, N. Feamster, A. G. Gray, and S. Krasser, "Detecting Spammers with SNARE: Spatio-temporal Network-level Automatic Reputation Engine," in *USENIX Security*, 2009.

[3] Z. Qian, Z. M. Mao, Y. Xie, and F. Yu, "On Network-level Clusters for Spam Detection," in *NDSS*, 2010.

[4] Y. Zhao, Y. Xie, F. Yu, Q. Ke, Y. Yu, Y. Chen, and E. Gillum, "BotGraph: Large Scale Spamming Botnet Detection," in *NSDI*, 2009.

[5] Y. Yu, M. Isard, D. Fetterly, M. Budiu, U. Erlingsson, P. K. Gunda, and J. Currey, "DryadLINQ: A System for General-Purpose Distributed Data-Parallel Computing Using a High-Level Language," in *OSDI*, 2008.

[6] H. Yu, M. Kaminsky, P. B. Gibbons, and A. Flaxman, "SybilGuard: Defending Against Sybil Attacks via Social Networks," in *SIGCOMM*, 2006.

[7] H. Yu, P. B. Gibbons, M. Kaminsky, and F. Xiao, "SybilLimit: A Near-Optimal Social Network Defense against Sybil Attacks," in *IEEE Symposium on Security and Privacy (Oakland)*, 2008.

[8] N. Tran, B. Min, J. Li, and L. Subramanian, "Sybil-Resilient Online Content Voting," in *NSDI*, 2009.

[9] B. Viswanath, K. P. Gummadi, A. Post, and A. Mislove, "An Analysis of Social Network-Based Sybil Defenses," in *SIGCOMM*, 2010.

[10] A. Mohaisen, A. Yun, and Y. Kim, "Measuring the Mixing Time of Social Graphs," in *IMC*, 2010.

[11] Y. Xie, F. Yu, Q. Ke, M. Abadi, E. Gillum, K. Vitaldevaria, J. Walter, J. Huang, , and Z. M. Mao, "Innocent by Association: Early Recognition of Legitimate Users," in *CCS*, 2012.

[12] Y.-Y. Ahn, S. Han, H. Kwak, S. Moon, and H. Jeong, "Analysis of Topological Characteristics of Huge Online Social Networking Services," in *WWW*, 2007.

[13] A. Bonato, J. Janssen, and P. Pralat, "A Geometric Model for On-line Social Networks," in *WOSN*, 2010.

[14] J. Leskovec, J. Kleinberg, and C. Faloutsos, "Graphs over Time: Densification Laws, Shrinking Diameters and Possible Explanations," in *KDD*, 2005.

[15] M. Sirivianos, K. Kim, and X. Yang, "SocialFilter: Introducing Social Trust to Collaborative Spam Mitigation," in *INFOCOM*, 2011.

[16] V. P. R. P. Oscar Boykin, "Leveraging Social Networks to Fight Spam," in *IEEE Computer Vol 38*, 2005.

[17] P. A. C. andJoerg Diederich and W. Nejdl, "MailRank: Global Attack-Resistant Whitelists for Spam Detection," in *CIKM*, 2005.

[18] S. Garriss, M. Kaminsky, M. J. Freedman, B. Karp, D. Mazieres, and H. Yu, "RE: Reliable Email," in *NSDI*, 2006.

[19] J. Huang, Y. Xie, F. Yu, Q. Ke, M. Abadi, E. Gillum, and Z. M. Mao, "SocialWatch: Detection of Online Service Abuse via Large-Scale Social Graphs," in *Tech Report MSR-TR-2013-24*, 2013.

[20] S. Brin and L. Page, "The Anatomy of a Large-Scale Hypertextual Web Search Engine," in *WWW*, 1998.

[21] Z. Bar-Yossef and L.-T. Mashiach, "Local Approximation of PageRank and Reverse PageRank," in *WWW*, 2008.

[22] A. D. Sarma, S. Gollapudi, M. Najork, and R. Panigrahy, "A Sketch-Based Distance Oracle for Web-Scale Graphs," in *WSDM*, 2010.

[23] E. Lehmann and J. P. Romano, *Testing Statistical Hypotheses*. Springer, 2005.

[24] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*. Wiley-Interscience, 2001.

[2] False negative accounts are those detected by the email provider but not by SocialWatch. Newly detected accounts are those classified as hijacked by SocialWatch but not by the email provider. They may actually be hijacked accounts that the current system fails to detect.