Design and Analysis of a Social Botnet

Yazan Boshmaf*, Ildar Muslukhov, Konstantin Beznosov, Matei Ripeanu

Department of Electrical and Computer Engineering, The University of British Columbia, 2332 Main Mall, Vancouver, B.C. Canada V6T 1Z4

Abstract

Online Social Networks (OSNs) have attracted millions of active users and have become an integral part of today's Web ecosystem. Unfortunately, in the wrong hands, OSNs can be used to harvest private user data, distribute malware, control botnets, perform surveillance, spread misinformation, and even influence algorithmic trading. Usually, an adversary starts off by running an infiltration campaign using hijacked or adversary-owned OSN accounts, with an objective to connect with a large number of users in the targeted OSN. In this article, we evaluate how vulnerable OSNs are to a large-scale infiltration campaign run by socialbots: bots that control OSN accounts and mimic the actions of real users. We adopted the design of a traditional web-based botnet and built a prototype of a Socialbot Network (SbN): a group of coordinated programmable socialbots. We operated our prototype on Facebook for eight weeks, and collected data about user behavior in response to a large-scale infiltration campaign. Our results show that (1) by exploiting known social behaviors of users, OSNs such as Facebook can be infiltrated with a success rate of up to 80%, (2) subject to user profile privacy settings, a successful infiltration can result in privacy breaches where even more private user data are exposed, (3) given the economics of today's underground markets, running a large-scale infiltration campaign might be profitable but is still not particularly attractive as a sustainable and independent business, (4) the security of socially-aware systems that use or integrate OSN platforms can be at risk, given the infiltration capability of an adversary in OSNs, and (5) defending against malicious socialbots raises a set of challenges that relate to web automation, online-offline identity binding, and usable security.

Keywords: Online social networks, Social network security, Automated social engineering, Online privacy, Botnets, Socialbots

1. Introduction

With more than a billion active users [1, 2], Online Social Networks (OSNs) such as Facebook[1] and Twitter[2] have attracted third parties who exploit them as effective online social media to reach out to and potentially influence a large and diverse population of web users [3, 4]. For example, OSNs were heavily employed by Obama's 2008 campaign team who raised about half a billion dollars online, introducing the digital era in presidential fundraising [5]. In addition, it has been argued that OSNs were one of the key enablers of the recent Arab Spring in the Middle East [6, 7]. This pervasive integration of OSNs into everyday life is rapidly becoming the norm, and arguably is here to stay [8]. Today's social Web, however, is threatened by cyber criminals who diligently attempt to attack many OSN platforms and breach the privacy of their users.

We recently showed that an adversary can infiltrate OSNs on a large scale by deploying an army of socialbots [9, 10]. A *socialbot* is an automation software that controls an adversary-owned or hijacked account on a particular OSN, and has the ability to perform basic activities such as posting a message and sending a connection request. What makes a socialbot different from self-declared bots (e.g., Twitter bots that post up-to-date weather forecasts) or *spambots* (i.e., bots that massively distribute unsolicited messages to non-consenting users) is that it is designed to pass itself off as a human being. This is achieved by either simply mimicking the actions of a real OSN user or by simulating such a user using artificial intelligence, just as in social robotics [11–13]. Thus, a socialbot can be used to *infiltrate* a targeted OSN in order to reach an influential position, that is, to compromise the *social graph* (i.e., the social structure) of the OSN by connecting with a large number of its users.

Large-scale infiltration in OSNs has serious security implications. First of all, a socialbot can pollute the targeted OSN with a large number of non-genuine social relationships. This means that it is unsafe to treat the infiltrated OSN as a trust network, which goes against the long-term health of the OSN ecosystem. In addition, third-party applications and websites have to perform the appropriate "clean up" in order to identify and remove most of the bogus user profiles along with their fake relationships, all before integrating or using such an OSN [14, 15].

Second, once a socialbot infiltrates a targeted OSN, it can exploit its new position in the network to spread misinformation in an attempt to bias the public opinion [16, 17], perform online surveillance [18], or even influence algorithmic trading that uses opinions extracted from OSNs to predict the stock market [19, 20]. For example, Ratkiewicz et al. [21] describe the use of Twitter bots to run astroturf campaigns during the

*Corresponding author. Tel: +1 604 822 2872. Fax: +1 604 822 5949.
Email addresses: `boshmaf@ece.ubc.ca` (Yazan Boshmaf), `ildarm@ece.ubc.ca` (Ildar Muslukhov), `beznosov@ece.ubc.ca` (Konstantin Beznosov), `matei@ece.ubc.ca` (Matei Ripeanu)

[1] `https://facebook.com`

[2] `https://twitter.com`

2010 U.S. midterm elections. Moreover, a socialbot can exploit its new position in the network to distribute malicious content such as botnet executables [22]. For example, the Koobface botnet [23] propagates by hijacking OSN accounts of infected machines, after which it uses these accounts to send messages containing a malicious link to other OSN users. When clicked, this link points to a legitimate but compromised website that attempts to infect its visitors with the Koobface malware.

Third and last, as a socialbot infiltrates the targeted OSN, it can also harvest private user data such as email addresses, phone numbers, and other personally identifiable information that have monetary value. To an adversary, such data are valuable and can be used for online profiling and large-scale email spam and phishing campaigns [24, 25]. It is thus not surprising that similar socialbots are being offered for sale in the Internet underground markets, with prices starting from $29 per single-featured bot and up to $2,500 per multi-featured bot [26].

A number of recently proposed techniques aim to automatically identify bots in OSNs based on their abnormal behavior [15, 27–30]. For example, Stein et al. [31] present the Facebook Immune System (FIS): an adversarial learning system that performs real-time checks and classification on every read and write action on Facebook's database, all for the purpose of protecting its users and the social graph from malicious activities. It is thus not surprising that such an adversarial learning system is rather effective at identifying and blocking spambots. Socialbots, on the other hand, are much more deceptive than spambots as they are designed to appear "normal" [9, 13, 32].

Graph theoretic techniques [14, 33, 34], as an alternative to adversarial learning systems, are expected to be less effective and more expensive at identifying socialbots, as one would typically "look for a needle in a haystack." Community detection algorithms [34, 35], for example, are deemed to fail as there will be far more fake relationships than socialbots [9, 14]. The intuition behind this is that each socialbot is expected to gradually, but independently, integrate into the targeted online community, resembling the scenario when a new user joins an OSN and starts connecting with others.

In this article, we extend our recent work on large-scale infiltration in OSNs [9, 10], and provide the first comprehensive study of this emerging threat that covers its human, economic, and technical factors. In particular, we enhance our treatment in tackling questions related to how OSN security defenses stand against socialbots that mimic real users, and how OSN users might behave in response to a large-scale infiltration campaign run by such deceptive bots. We also provide new results related to how much leverage an adversary might gain by running a large-scale infiltration campaign, whether it is economically feasible to run such a campaign in the first place, and what the expected challenges faced by OSN security defenses might be.

We studied large-scale infiltration in OSNs as an organized campaign run by an army of socialbots to connect with either random or targeted OSN users on a large scale. We adopted the design of a traditional web-based botnet and defined what we call a *Socialbot Network* (SbN): a group of programmable socialbots that are coordinated by an adversary (referred to as a *botherder*) using a software controller (referred to as a *botmaster*). We designed the botmaster to exploit the known properties of social networks, such as the *triadic closure principle* [36], in order to improve the magnitude of the potential infiltration.

We created a fairly small and simple, yet effective, SbN consisting of 102 socialbots and a single botmaster, and then operated this SbN on Facebook for eight weeks. During that time, the socialbots sent a total of 8,570 friendship requests, out of which 3,055 were accepted. We recorded all data related to the resulted infiltration by this SbN and the corresponding user behavior, along with all accessible user profile information. Our findings can be summarized as follows:

- OSNs such as Facebook are vulnerable to large-scale infiltration campaigns.

From the OSN side, we show that today's OSNs exhibit inherent vulnerabilities that allow an adversary to automate the infiltration on a large scale (Section 3), and accordingly, build an SbN that is difficult to detect by OSN security defenses such as the FIS (Section 4). From the user side, we show that most OSN users are not careful enough when accepting friendship requests, especially when they share mutual friends with the sender. This behavior can be exploited to achieve a large-scale infiltration with a success rate of up to 80% (Sections 5 and 6).

- Depending on user profile privacy settings, operating an SbN can result in serious privacy breaches.

We show that after a large-scale infiltration, a botherder can harvest large amounts of publicly inaccessible user data. This data include email addresses, phone numbers, and other profile information of the infiltrated users, all of which have monetary value. Unfortunately, this also includes the private data of "friends of friends", that is, users who have not been infiltrated but are friends with infiltrated users (Section 6).

- Operating an SbN is expected to be profitable, but it is not particularly attractive as an independent business.

We analyzed the economic feasibility of operating an SbN, and based on a simple cost-volume-profit analysis, we show that a botherder can make profit by either selling harvested user data or earning a lump-sum payment for infiltrating a predetermined number of OSN users. Our analysis, however, indicates that large-scale infiltration is not sustainable as an independent business, and a rational botherder would utilize an SbN as a tool to collect private user data in order to personalize subsequent, more profitable adversarial campaigns such as email-based spam and phishing campaigns (Section 7).

- The security of socially-aware software systems can be at risk, given the infiltration capability of an adversary.

We show that it is *not* difficult for an adversary to establish arbitrarily many social connections with arbitrary users in OSNs such as Facebook. The opposite, however, is often assumed by software systems that use the social graph of OSNs as a trust network in order to provide socially-aware services such as personalized collaborative filtering and Sybil defense in peer-to-peer systems. We use Distributed Hash Tables (DHTs) as an example, and demonstrate that this assumption is unsafe and can lead to the situation where an adversary circumvents the employed security protocol in order to compromise the functionality of the DHT (Section 8).

- OSN security defenses have a higher leverage detecting infiltration campaigns, rather than trying to prevent them.

Our investigation suggests that preventing large-scale infiltration in OSNs is a hard problem, as it involves fixing inherent vulnerabilities found in today's OSNs. In fact, this boils down to solving a set of socio-technical challenges that relate to web automation, online-offline identity binding, and usable security. We believe that OSN defenses have a higher leverage in detecting, and thereof, limiting the implications of large-scale infiltration campaigns, especially when such systems are iteratively updated to mitigate adversarial attacks. Doing so has the effect of increasing the cost of running large-scale adversarial campaigns and shrinking the corresponding profit margins, which eventually render such campaigns economically infeasible and non-scalable (Section 9).

Our findings are not limited to Facebook. Recent research shows that other OSNs such as Twitter are vulnerable to large-scale infiltration campaigns as well [13, 21, 32, 37, 38], which, unfortunately, exposes a population of more than one billion OSN users to this emerging threat.

In conclusion, our findings shed light on the importance of considering the human factor when designing OSN security defenses. We believe that socio-technical solutions are required to effectively protect the social Web and realize security defenses that are less vulnerable to both human and technical exploits (i.e., social engineering and OSN platform hacks, respectively).

2. Background and Preliminaries

In what follows, we present background information and define the notations we use in the upcoming discussion.

2.1. Online Social Networks

An *Online Social Networks* (OSN) is a centralized web *platform* that facilitates and mediates user's social activities online. A user in such a platform owns an account and is represented by a profile that describes her social attributes such as name, gender, interests, and contact information. We use the terms "account", "profile", and "user" interchangeably but make the distinction explicit when deemed necessary. A social relationship between two users can be either undirected such as friendships in Facebook, or directed such as followerships in Twitter.

An OSN can be modeled as a *social graph* $G = (V, E)$, where V represents a set of users and E represents a set of social connections (i.e., relationships) among these users. For every user $u \in V$, the set $\Gamma(u)$ is called the *neighborhood* of u, and it contains all users in V with whom u has social connections. We denote the *average neighborhood size* in G by $\overline{N}(G) = \eta$, which is defined as follows:

$$\overline{N}(G) = \frac{1}{|V|} \sum_{u \in V} |\Gamma(u)| = \eta. \tag{1}$$

Finally, we call the set $\Delta(u)$ the *extended neighborhood* of u, which is defined as the union of the neighborhoods of all users in $\Gamma(u)$ as follows:

$$\Delta(u) = \bigcup_{v \in \Gamma(u)} \Gamma(v). \tag{2}$$

For example, in Facebook, $\eta = 130$ and the sets $\Gamma(u)$ and $\Delta(u)$ represent the "friends" and the "friends of friends" of user u, respectively [1].

2.2. Sybil Attacks and Infiltration in OSNs

The *Sybil attack* [39] refers to the situation where an adversary controls multiple identities, each called a *Sybil*, and joins a targeted system under these identities many times in order to subvert a particular service. Accordingly, we define *large-scale infiltration* in OSNs as an instance of the Sybil attack where an adversary employs an automation software, which is scalable enough to control many adversary-owned or hijacked accounts (i.e., Sybils), in order to connect with a large number of users in the targeted OSN (i.e., the targeted system).

Recent research indicates that large-scale infiltration in OSNs is possible [40–42]. For example, Bilge et al. [43] show that most users in OSNs are not cautious when accepting connection requests that are sent to them. The authors performed an experiment on Facebook to test how willing users are to accept friendship requests sent from forged user profiles of people who were already in their friends list as confirmed contacts. They also compared that with users' response to friendship requests sent by people they do not know (i.e., fake profiles representing strangers). In their experiment, they show that the acceptance rate for forged profiles was always over 60%, and about 20% for the fake profiles. Unlike their targeted attack, we do not expect the adversary to forge user profiles as this makes the attack non-scalable and more susceptible to detection [31]. Moreover, we aim to characterize more descriptive user behaviors that are important to improve today's OSN security defenses, and to evaluate the corresponding security and privacy implications, all under the context of large-scale infiltration. To the best of our knowledge, we present the first comprehensive treatment of this emerging OSN security topic.

2.3. Social Engineering, Automation, and Socialbots

Traditionally, *social engineering* is defined as the art of gaining access to secure objects by exploiting human psychology, rather than using hacking techniques [44]. Social engineering has become more technical and complex; social engineering attacks are being computerized and fully automated, and are becoming adaptive and context-aware [18, 23, 24, 42, 43, 45].

Huber et al. [46] presented one of the first frameworks for automated social engineering in OSNs, where a new breed of bots can be used to automate traditional social engineering attacks for many adversarial objectives. Given this context, a *malicious socialbot* can be thought of as an automated social engineering tool that allows an adversary to infiltrate online communities like a virtual con man, build up trust over time, and then exploit it to elicit information, sway opinions, and call to action.

In fact, automation has a strong economic rationale behind it. Herley [47] shows that for an online attack to be scalable, it ought to be automated without manual per-user adjustments. Otherwise, there are no economic incentives for a *rational* (i.e., profit-driven) adversary to scale the attack, which is undesirable from an adversarial standpoint.

Socialbots can be used for non-adversarial objectives as well [13]. For example, The Web Ecology Project [48] envisions the design of *benign socialbots* that have positive impact on online communities, where bots are used to advocate awareness and cooperation among human users on civic or humanitarian issues. Soon after, this objective got extended towards realizing online *social architecture* [37]: the technology where "intelligent" socialbots are used to interact with, promote, and provoke online communities towards desirable behaviors, including large-scale restructuring of social graphs.

3. OSN Vulnerabilities

We discuss four vulnerabilities found in today's OSNs that allow an adversary to run a large-scale infiltration campaign. Collectively, along with poorly designed end-user privacy controls [49], these vulnerabilities represent the *enabling factors* that make operating socialbots feasible in the first place.

3.1. Ineffective CAPTCHAs

OSNs employ CAPTCHAs [50] to prevent automated bots from abusing their platforms. An adversary, however, can often circumvent this countermeasure by using different techniques such as automated analysis via optical character recognition and machine learning [43], exploiting botnets to trick the infected victims into manually solving CAPTCHAs [23, 51], reusing session IDs of known CAPTCHAs [52], cracking MD5 hashes of CAPTCHAs that are validated on the client side (i.e., the web browser) [53], or hiring cheap human labor [54].

Let us consider the use of human labor to solve CAPTCHAs, which is a phenomenon known as CAPTCHA-solving business. Motoyama et al. [54] show that companies involved in such a business are surprisingly efficient: they have high service quality with a success rate of up to 98%, charge \$1 per 1,000 successfully solved CAPTCHAs, and provide software APIs to automate the whole process. Thus, even the most sophisticated CAPTCHA that only humans could solve can be effectively circumvented with a small investment from an adversary. In such a situation, the adversary acts as an economist and invests in such businesses if the expected return on investment is considerably high. This allows researchers to study online attacks from an economic context, and define cost metrics and structures that measure when it is economically feasible for an adversary to mount a large-scale attack that involves, for instance, solving CAPTCHAs by employing cheap human labor [47]. We provide such an analysis for large-scale infiltration in Section 7.

3.2. Sybil Accounts and Fake Profiles

Creating a user account on an OSN involves three tasks: providing an active email address, creating a user profile, and sometimes solving a CAPTCHA. Each user account maps to one profile, but many user accounts can be owned by the same person or organization using different email addresses. The latter case represents a potential Sybil attack, which we further study in Section 8. In what follows, we show that an adversary can fully automate the account creation process in order to create a set of Sybil user accounts, where each account is represented by a fake user profile. This, however, is not new as similar tools are used for online marketing [55]. The adversary can write a customized software to create such accounts or buy OSN accounts in bulk online [56, 57].

3.2.1. Sybil User Accounts

When creating a new user account on an OSN, an email address is required to first validate and then activate the account. The OSN validates the account by associating it with the owner of the email address. After account validation, its owner activates the account by following an activation link that is emailed by the OSN. Accordingly, an adversary has to overcome two hurdles when creating a new Sybil account: providing an active email address that he owns and account activation. To tackle the first hurdle, the adversary can maintain many email addresses by either using "temp" email addresses that are obtained from providers that do not require registration such as 10MinuteEmail[3], or by creating email addresses using email providers that do not limit the number of created email accounts per browsing session or IP address such as MailRu[4]. As for the second hurdle, an adversary can write a simple script that downloads the activation email and then sends an HTTP request to the activation URL, which is typically included in the downloaded email.

3.2.2. Fake User Profiles

Creating a user profile is a straightforward task for real users as they just have to provide the information that represents their social attributes. For an adversary, however, the situation is different. The objective of the adversary is to create profiles that are "socially attractive". We consider a purely adversarial standpoint concerning social attractiveness where the adversary aims to exploit certain social attributes that have shown to be effective in getting users' attention. Such attributes can be inferred from recent social engineering attacks. Specifically, using a profile picture of a good looking woman or man has had the greatest impact [43, 58]. Thus, an adversary can use publicly available personal pictures for the newly created profiles, with the corresponding gender and age range information. The adversary can use already rated personal pictures from websites like HotOrNot[5], where users publicly post their personal pictures for others to rate their "hotness". In fact, such websites also provide categorization of the rated personal pictures based on gender and age range. It is thus possible for an adversary to automate the collection of the required profile information in

[3]`http://10minutemail.com`
[4]`http://mail.ru`
[5]`http://hotornot.com`

order to populate a fake user profile by crawling, or scavenging in this case, the Web.

3.3. *Crawlable Social Graphs*

The social graph of an OSN is usually hidden from public access in order to protect its users' privacy. An adversary, however, can reconstruct parts of the social graph by first logging in to the OSN platform using one or many accounts, and then traversing through linked user profiles starting from a seed profile. In the second task, web crawling techniques can be used to download profile pages and then scrape their content. This allows the adversary to parse the connections lists of user profiles, such as the "friends list" in Facebook, along with their profile information. After that, the adversary can gradually construct the corresponding social graph with all accessible social attributes using an online search algorithm such as breadth-first search [59]. The adversary can build either a customized web crawler for this task or resort to cheap commercial crawling services that support social-content crawling such as 80Legs[6].

3.4. *Exploitable Platforms and APIs*

Most OSNs provide software APIs that enable the integration of their platforms into third-party software systems. Facebook Graph API [60], for example, enables third parties to read from and write data into Facebook, and provides a simple and consistent view of Facebook's social graph by uniformly representing objects (e.g., profiles, photos) and the connections between them (e.g., friendships, likes, tags). An adversary, however, can use such APIs to automate the execution of social activities online. If an activity is not supported by the API, then the adversary can scrape the content of the platform's website, and record the exact HTTP requests which are used to carry out such an activity (i.e., HTTP-request templates). In particular, sending connection requests is often not supported, and is usually protected against automated usage by CAPTCHAs. This is also the case if a user sends too many requests in a short time period. An adversary, however, can always choose to reduce the frequency at which he sends the requests to avoid CAPTCHAs. Another technique is to inject artificial connection requests into normal OSN traffic at the HTTP level, so that it would appear as if the users added the adversary as a friend [61].

4. The Socialbot Network

We first start with a conceptual overview of a Socialbot Network (SbN) and its threat model. This is followed by a discussion on the SbN design requirements, after which we outline its construction details.

4.1. *Overview*

We define a *Socialbot Network* (SbN) as a set of socialbots that are owned and maintained by a human controller called the *botherder* (i.e., the adversary). As shown in Figure 1, an

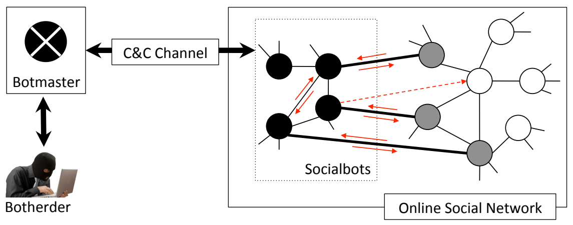


Figure 1: A Socialbot Network (SbN). Each node in the OSN represents a user profile. The socialbots are marked in black. Infiltrated users are marked in gray. Edges between nodes represent social connections. The dashed arrow represents a connection request. The small arrows represent social interactions. The SbN can be part of an existing botnet, where each "zombie" machine is additionally infected by the socialbot malware that controls a fake user profile in the targeted OSN.

SbN consists of three components: socialbots, a botmaster, and a Command & Control (C&C) channel. Each socialbot controls a profile in a targeted OSN, and is capable of executing commands that result in operations related to social interactions (e.g., posting a message) or the social structure (e.g., sending a connection request). These commands are either sent by the botmaster or predefined locally on each socialbot. All data collected by the socialbots are called the *botcargo*, and are always sent back to the botmaster. A *botmaster* is an OSN-independent software controller with which the botherder interacts in order to define commands that are sent through the C&C channel. The C&C *channel* is a communication medium that facilitates the transfer of the botcargo and the commands between the socialbots and the botmaster, including any heartbeat signals.

4.2. *Threat Model*

We assume the threat model of a global passive adversary who is capable of building and operating a fully- or semi-automated SbN on a large scale. This involves exploiting all of the vulnerabilities presented in Section 3, which collectively enable the operation of such an SbN in the targeted OSN. We also assume that the adversary is capable of deploying the SbN as part of an existing botnet, and thus, we treat the SbN as a distributed network of compromised "zombie" machines acting cooperatively. Accordingly, we believe it is fair to assume that the defenders (i.e., OSNs and ISPs) are able to cooperate, and hence, have a global view of the communication traffic. Finally, we assume that botnet infections are not easily detected, that is, an SbN cannot tolerate 100% clean up of all infected machines, just like any other botnet. We expect, however, an SbN to tolerate random losses of a large number of compromised machines because at least one machine is required to host all of the socialbots, as we show in Section 5.

As for the adversarial objectives, the botherder builds and operates an SbN to (1) carry out a large-scale infiltration campaign in a targeted OSN, and (2) harvest private user data. The first objective involves connecting with a large number of either random or targeted OSN users for the purpose of establishing an influential position, which can be then exploited to promote malicious content or spread misinformation. The second objective, on the other hand, aims to generate profit by collecting

[6]http://80legs.com

Table 1: The generic operations supported by a socialbot in any given OSN.

Operation	Type	Description
`read`(o, p)	Social-interaction	Reads object o from profile p and returns its value v as botcargo
`write`(v, o, p)	Social-interaction	Writes value v to object o on profile p
`connect`(b, p)	Social-structure	Sends or accepts a connection request sent from profile b to profile p
`disconnect`(b, p)	Social-structure	Breaks the social connection between profiles b and p

personal user data that have monetary value. In addtion, the data can be used to craft personalized messages for subsequent spam, phishing, or astroturf campaigns.

4.3. *Design Requirements*

Ideally, an SbN has to be automated and scalable enough to control hundreds of socialbots. This is achieved by adopting the design of a traditional web-based botnet. In order to be effective, however, an SbN has to meet three challenging requirements: (1) each socialbot has to be designed in such a way that hides its true face (i.e., a robot), (2) the botmaster has to implement heuristics that enable large-scale infiltration in the targeted OSN, and (3) the traffic in the C&C channel has to look benign in order to avoid detecting the botmaster.

In this article, we decided to use a simplistic design in order to meet each one of these requirements. We used techniques that have shown to be both feasible and effective. We acknowledge, however, that more sophisticated techniques that utilize machine learning algorithms are possible. We refrain from using such techniques as our objective is to evaluate the threat of large-scale infiltration and characterize user behavior, rather than to optimize the performance of an SbN. We discuss the details of the used techniques in the following section.

4.4. *Construction*

We now discuss how a botherder can construct an SbN that meets the design requirements and performs well in practice.

4.4.1. *The Socialbots*

A socialbot consists of two main components: a profile on a targeted OSN (the face), and the socialbot software (the brain). Given that we develop the socialbot software in an adverserial setting, we regard this software as being malicious, and refer to it as malware. We enumerate the socialbots with the profiles they control, that is, for a set $\mathcal{B} = \{b_1, \ldots, b_m\}$ of m socialbots, we use $b_i \in \mathcal{B}$ to refer to both the i-th socialbot and the profile it controls. But how should the socialbot malware be programmed in order to mimic real users?

First, we require the socialbot to support two types of *generic operations* in any given OSN: (1) social-interaction operations that are used to read and write social content, and (2) social-structure operations that are used to alter the social graph. A description of these operations is shown in Table 1.

Second, we define a set of *commands* that each includes a sequence of generic operations. A command is used to mimic a real user action that relates to social content generation (e.g., a status update) or social networking (e.g., joining a community of users). Commands can be either defined locally on each socialbots (referred to as *native commands*), or sent by the botmaster to the socialbots through the C&C channel (referred to as *master commands*). For example, we define a native command called `status_update` as follows: at arbitrary times, a socialbot $b_i \in \mathcal{B}$ generates a message m (e.g., a random blurb crawled from the Web), and executes the operation `write`(m, o, b_i) where o is the object that maintains messages on profile b_i (e.g., the profile's "wall" in Facebook). This command resembles an OSN user posting a status update message on her profile, and is executed at arbitrary times in order to avoid creating detectable patterns. Likewise, more sophisticated commands can be defined that, for instance, allow the socialbots to comment on each others' status updates.

Third, each socialbot can be enhanced with advanced social-interaction capabilities by integrating existing chatterbots into the socialbot's malware [46, 62], or by hijacking online human-to-human conversations in a man-in-the-middle fashion [63].

Finally, each socialbot employs a *native controller*: a simple two-state Finite-State Machine (FSM) that enables the socialbot to either socialize by executing commands or stay dormant.

4.4.2. *The Botmaster*

A botmaster is a botherder-controlled automation software that orchestrates the overall operation of an SbN. The botmaster consists of three main components: a botworker, a botupdater, and a C&C engine. The *botworker* builds and maintains socialbots. Building a new socialbot involves first creating a new socially attractive user profile in the targeted OSN as discussed in Section 3.2. After that, the profile's credentials (i.e., the user name and password) are delegated to the socialbot's malware in order to get a full control over this profile. If the SbN is operated as part of a botnet, the socialbot malware can use hijacked OSN accounts instead. This, however, might make the socialbot more susceptible to detection [23]. The *botupdater* pushes new software updates, such as new native commands or updated HTTP-request templates, to the socialbots through the C&C channel. Finally, the C&C *engine* maintains a repository of master commands and runs a *master controller*: a many-state FSM that is the core control component of the SbN. The botherder interacts with the C&C engine to define a set of master commands, which are dispatched when needed by the master controller and then sent to the socialbots. Two interesting questions now follow: (1) what kinds of master commands are required to achieve a large-scale infiltration in the OSN, and (2) when should they be dispatched by the master controller?

First, notice that at the beginning each socialbot is isolated from the rest of the OSN, that is, $|\Gamma(b_i)| = 0$ for each $b_i \in \mathcal{B}$, which is not a favorable structure to start a large-scale infiltration. Tong et al. [64] show that the social attractiveness of a

Table 2: Master commands. The socialbot $b_i \in \mathcal{B}$ is the socialbot executing the command, where $|\mathcal{B}| = m$ and $\bar{N}(G) = \eta$.

Command	Description
`cluster`	Connects b_i with at most η other socialbots in $\mathcal{B}$
`rand_connect(`k`)`	Connects b_i with k non-boherder-owned profiles that are picked at random from G
`decluster`	Disconnects b_i from every socialbot $b_j \in \mathcal{S}$ where $\mathcal{S} = \{b_j \mid b_j \in \Gamma(b_i) \cap \mathcal{B}$ and $|\Gamma(b_j)| > m\}$
`crawl_extneighborhood`	Returns $\Delta(b_i)$, the extended neighborhood of b_i, as botcargo
`mutual_connect`	Connects b_i with every profile $p_j \in \Delta(b_i) - \mathcal{B}$.
`harvest_data`	Reads all accessible information of every profile $p_j \in \Gamma(b_i)$, and returns it as botcargo

profile in an OSN is highly correlated to its neighborhood size, where the highest attractiveness is observed when the neighborhood size is close to the network's average $\overline{N}(G) = \eta$. Usually, η is known or can be estimated. Thus, in order to increase the social attractiveness of a socialbot, the botherder defines a master command `cluster`, which orders each socialbot to connect with at most η other socialbots. Moreover, this might be helpful in order to elude OSN security defenses that keep track of how many rejected connection requests each new user ends up with when joining the OSN for the first time, which is usually used as an indication of an automated activity or spam [30].

Second, it has been widely observed that if two users have a mutual connection in common, then there is an increased likelihood that they become connected themselves in the future [65]. This property is known as the *triadic closure principle* [36], and it originates from real-life social networks. Nagle et al. [40] show that the likelihood of accepting a connection request in an OSN is about three times higher given the existence of some number of mutual connections. Therefore, in order to improve the potential infiltration in the targeted OSN, the botherder defines a master command `mutual_connect`, which orders each socialbot to connect with users with whom it has some mutual connections (i.e., users in the extended neighborhood $\Delta(b_i)$ for each socialbot $b_i \in \mathcal{B}$).

Finally, we designed the master controller to switch between three super states or phases: setup, bootstrapping, and propagation. In the *setup* phase, the botmaster builds m socialbots, updates their malware, and then issues the `cluster` command. After that, in the *bootstrapping* phase, the botmaster issues the command `rand_connect(`k`)`, which orders each socialbot to connect with k user profiles that are picked at random from the targeted OSN. When every socialbot is connected with k non-botherder-owned profiles, the botmaster issues the command `decluster`, which orders the socialbots to break the social connections between them, and hence, destroying any m-clique structure that could have been created in the earlier step. In more advanced implementations, a socialbot would break one social connection with another socialbot for every newly infiltrated user profile, and thus, it gradually `declusters`. In the *propagation* phase, the botmaster issues the command `crawl_extneighborhood`, which orders the socialbots to crawl their extended neighborhoods, after which the botmaster uses the crawled information and issues the command `mutual_connect`. Whenever a socialbot infiltrates a user profile, the botmaster issues the command `harvest_data`, which orders the socialbot to collect all accessible user profile information in its neighborhood and return it as a botcargo. A description of all master commands is shown in Table 2.

4.4.3. The C&C Channel

The communication model of an SbN consists of two channels: the C&C channel and the socialbot-OSN channel. The socialbot-OSN channel carries only OSN-specific API calls and normal HTTP traffic, which are the end product of executing a command by a socialbot. From the OSN side, this traffic originates from either an HTTP proxy in case of high activity, or from a normal user. It is therefore quite difficult to identify a socialbot solely based on the traffic it generates in the socialbot-OSN channel.

As for the C&C channel, how should it be designed so that it is particularly hard to identify the botmaster? To start with, we argue that detecting the botmaster from the C&C traffic is as hard as it is in a traditional botnet because the botherder can rely on an existing botnet infrastructure and deploy the SbN as part of the botnet, as discussed in Section 4.2. Alternatively, the botherder can exploit the OSN platform itself for the C&C infrastructure [66]. For example, Nagaraja et al. [67] show that a botherder can establish a probabilistically unobservable communication channel by building a covert OSN botnet.

5. Evaluation

In order to evaluate how vulnerable OSNs are to a large-scale infiltration by an SbN, we decided to build one according to the discussion in Section 4.4. We chose Facebook as the targeted OSN because it is the largest OSN found today, consisting of more than 750 million users [1]. Besides, we believe it is particularly difficult to operate an SbN on Facebook as (1) unlike other OSNs, Facebook is mostly used to connect with real-life friends and family but not with strangers [68–70], and (2) Facebook employs the Facebook Immune System (FIS) [31]: a real-time adversarial learning system which represents a potential nemesis of any SbN.

5.1. Ethics Consideration

Given the nature of an SbN, is it ethically acceptable and justifiable to conduct such a research experiment? We believe that controlled, minimal-risk, and realistic experiments are the only way to reliably estimate the feasibility of an attack in real-world. These experiments allow us and the wider research community to get a genuine insight into the ecosystem of online attacks, which is useful in understanding how similar attacks may behave and how to defend against them.

We carefully designed our experiment in order to reduce any potential risk at the user side [71]. In particular, we followed the known practices and got the approval of our university's behavioral research ethics board. We strongly encrypted and properly anonymized all collected data, which we have completely deleted after we finished our planned data analysis.

As part of our code of ethics, we communicated the details of our experiment to Facebook before any publication, and accordingly, we decided not to include specific technicalities about a set of vulnerabilities we discovered in Facebook's platform. We believe that these platform vulnerabilities can be exploited by cyber criminals to mount different kinds of online attacks. We reported these vulnerabilities to Facebook through its platform's online vulnerability reporting tool [72].

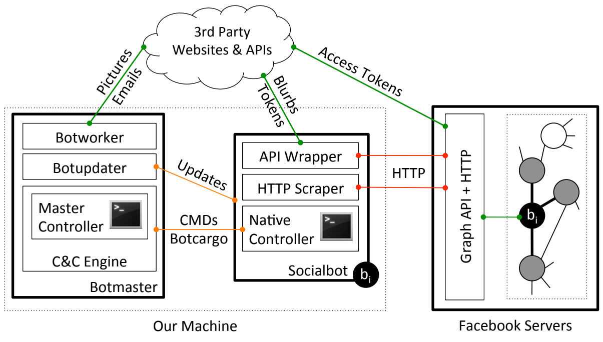


Figure 2: The Facebook Socialbot Network.

5.2. Methodology

Our main research objective is to characterize users' response to a large-scale infiltration campaign in OSNs, along with the corresponding security and privacy implications. To this end, we built an SbN prototype targeting Facebook for the reasons outlined above, and operated this SbN for eight weeks during the first quarter of 2011. The duration of the experiment was informed by how many data points we needed to properly capture user behavior, and accordingly, we took the SbN down once we stopped observing new trends. We report only the results we observed during the length of the experiment. We used a single machine and two types of IP addresses at different stages of the experiment. The first IP address was assigned by the university, and the second IP address was assigned by a commercial ISP. We also implemented a simple HTTP proxy on the machine we used in order to make the traffic looks like as if it originated from multiple clients having different browsers and operating systems. Even though the university-assigned IP address might have diluted the Facebook Immune System [73], we believe that it is unsafe to completely white-list university IP addresses.[7] In fact, today's botnet owners struggle over who has the largest number of "high-quality" infected machines, including university, corporate, and even government machines [74].

5.3. The Facebook SbN

Figure 2 shows the architecture of the SbN we developed. Each socialbot ran the same malware and was equipped with only one native command; `status_update`. We implemented the generic operations described in Table 1 using two techniques: API calls and HTTP-request templates, which we now briefly describe. First, we exploited Facebook's Graph API [60] to carry out the social-interaction operations. The API, however, requires the user (i.e., the socialbot in this case) to be logged in to Facebook at the time of any API call. To avoid this, we developed a Facebook application that fetches permanent OAuth 2.0 access tokens in order to allow each socialbot to send API calls without the need to login. Second, for the social-structure operations, we used pre-recorded HTTP-request templates that allow each socialbot to send friendship requests as if they were sent from a browser. We used an API provided by iHeartQuotes[8] to pull random quotes and blurbs, and used them as messages for the status updates. As for the botmaster software, we implemented the botworker to interface with three useful websites: DeCaptcher[9], a CAPTCHA-solving business; HotOrNot, a photo-sharing website; and MailRu, an email provider. We also implemented the botupdater with an enhanced functionality to update any particular HTTP-request template, along with any new native commands. Finally, we implemented all master commands described in Table 2.

The master command `rand_connect(`k`)` requires some extra attention. On Facebook, each user profile has a unique identifier that is represented by a 64-bit integer and is assigned at the time the profile is created. In order to get a uniform random sample of Facebook profiles, we decided to use a simple random sampling technique called *rejection sampling* [75], which we now describe. First, we generated 64-bit integers at random, but with a range that is reduced to the known identifier ranges used by Facebook [76]. Next, we tested whether each newly generated identifier mapped to an existing user profile by probing the profile's page using this identifier. Finally, if the profile existed, we included this profile identifier in the random sample only if this profile was not isolated. We define an *isolated* user profile as a profile that does not display the user's "friends list" or has no friends of Facebook.

We deployed the simple two-state native controller and the three-phase, many-state master controller. A more resourceful attacker, however, would employ adversarial classifier reverse engineering techniques [77] in order to learn sufficient information about the security defenses deployed by the targeted OSN, and then construct an adversarial attack that maximizes the potential infiltration and minimizes the detection rate.

[7]During the SbN operation using a university IP address, we observed that some actions were identified as malicious, and the used IP address was temporarily blocked, especially during Sybil accounts creation. This supports the argument that even university IP addresses were audited by Facebook, and they were not fully white-listed.

[8]`http://iheartquotes.com`
[9]`http://decaptcher.com`

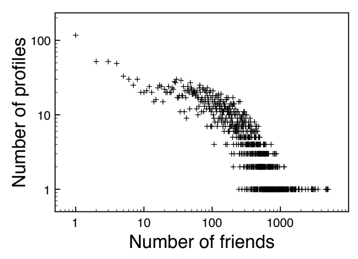


Figure 3: Degree distribution of the generated random sample of Facebook user profiles during the bootstrapping phase, with a sample size of 5,053 valid profile identities.

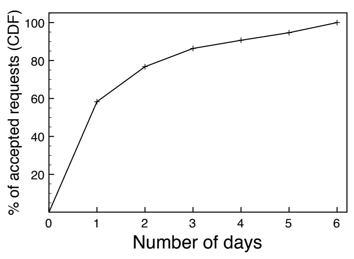


Figure 4: Cumulative distribution function of number of days it took to observe a fraction of the overall accepted friendship requests during the bootstrapping phase.

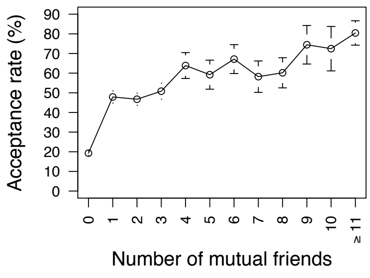


Figure 5: Average acceptance rate of the resulted infiltration as a function of the number of mutual friends the socialbots had with the infiltrated users. (95% conf.)

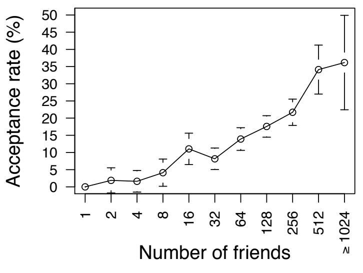


Figure 6: Average acceptance rate as a function of the number of friends a user profile has during the bootstrapping phase. (for the requests sent by *m*-socialbots, 95% conf.)

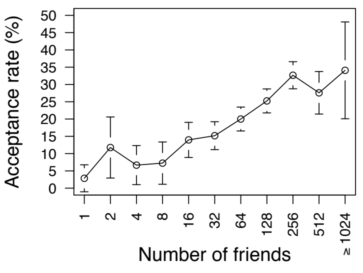


Figure 7: Average acceptance rate as a function of the number of friends a user profile has during the bootstrapping phase. (for the requests sent by *f*-socialbots, 95% conf.)

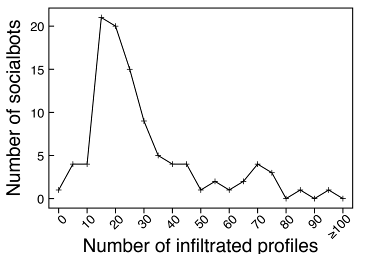


Figure 8: Distribution of the overall infiltration over the socialbots. A point in the figure represents how many socialbots infiltrated the corresponding number of profiles.

5.4. Operating the Facebook SbN

We operated the Facebook SbN for eight weeks during the first quarter of 2011. The socialbots were able to send a total of 8,570 friendship requests, out of which 3,055 requests were accepted by the infiltrated users. We divide the following discussion according to the three phases of the master controller.

5.4.1. Setup

As $\eta = 130$ on Facebook, we decided to create 102 socialbots and a single botmaster, all of which are physically hosted on a single machine for simplicity. A botherder, however, would resort to a more sophisticated deployments such as peer-to-peer overlay networks [78]. Even though we could have built the socialbots automatically using the botworker, we decided to create them manually as we had no intention to support any CAPTCHA-solving business. In total, we created 49 socialbots that had male user profiles, to which we refer as *m*-socialbots, and 53 socialbots that had female user profiles, to which we refer as *f*-socialbots. As expected, the socialbots clustered into a 102-clique structure, representing a tightly-knit, cohesive community of OSN users that is useful for bootstrapping the infiltration, as discussed in Section 4.4.2

5.4.2. Bootstrapping

The socialbots generated a random sample of 5,053 valid user profile identities. These unique identities passed the inclusion criteria we presented in Section 5.3. Figure 3 shows the degree distribution of this uniform sample.[10]

Based on a pilot study, we decided to send 25 friendship requests per socialbot per day in order to avoid CAPTCHAs. The socialbots took two days to send friendship requests to all of the 5,053 profiles. In total, exactly 2,391 requests were sent from *m*-socialbots and 2,662 from *f*-socialbots. We kept monitoring the status of the requests for six days. Overall, 976 requests were accepted with an average acceptance rate of 19.3%. In particular, 381 of the accepted requests were sent from *m*-socialbots (15.9% acceptance rate), and 595 were sent from *f*-socialbots (22.3% acceptance rate). The difference in the average acceptance rate was statistically significant ($\chi^2 = 32.8702$, $p = 9.852 \times 10^{-9}$), where the *f*-socialbots outperformed the *m*-socialbots by 6.4% on average.[11] Approximately 86% of the infiltrated users accepted the requests within the first three days of the requests being sent, as shown in Figure 4. In our implementation, the socialbots gradually broke the 102-clique structure as they infiltrated more and more user profiles. Overall, the SbN spent two weeks in the bootstrapping phase. For most of that time, however, the SbN was setting idle.

5.4.3. Propagation

We kept the SbN running for another six weeks. During that time, the socialbots added 3,517 more user profiles from their

[10]The *degree* of a node is the size of its neighborhood, and the *degree distribution* is the probability distribution of these degrees over the whole network or a sample of it.

[11]Using a 2-sample test for equality of proportions.

extended neighborhoods, out of which 2,079 profiles were successfully infiltrated. This resulted in an average acceptance rate of 59.1%, which, interestingly, depends on how many mutual friends the socialbots had with the infiltrated users, and can increase up to 80% as shown in Figure 5.

By the end of the eighth week, we decided to take the SbN down as we stopped observing new trends in user behavior. Moreover, the SbN resulted in a heavy traffic with Facebook where we recorded approximately 250GB inbound and 3GB outbound of network traffic. We consider the operation time a conservative estimate of the real performance of the SbN as we paused it several times for debugging and data analysis, especially during the bootstrapping phase. For example, Facebook changed the HTTP-request parameters used for sending a friendship request through their platform, and thus, we had to pause the SbN operation in order to record the new parameters, use the botupdater to push the new HTTP-request template to the socialbots, and then continue the SbN operation. We believe that operating the SbN for a longer time is expected to increase the average acceptance rate as the propagation phase will have a higher contribution, as suggested by Figure 5.

6. Discussion

In what follows, we discuss the results presented in the previous section and focus on three main points: the observed user behavior, the harvested user data, and the infiltration performance of the socialbots.

6.1. User Behavior

Given the results presented in Section 5, are the infiltrated users real after all, or are they just other socailbots? To begin with, notice that during the bootstrapping phase, the socialbots targeted profiles that were picked at random out of hundreds of millions of user profiles, and thus, it is highly unlikely to have picked mostly socialbots.[12]

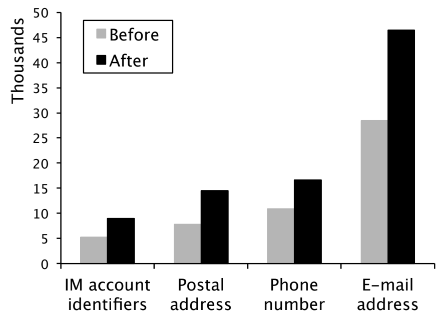
We also support this argument by the following analysis of the observed user behavior. First of all, consider Figure 5. The big jump in the acceptance rate from users who were picked at random to those with whom the socialbots had some mutual friends is expected. It directly exhibits the effect of the triadic closure principle, which predicts that having mutual friends would increase the likelihood of accepting a friendship request, as discussed in Section 4.4.2. In fact, this resulted in the following correlation: the more mutual friends a socialbot had with a user, the higher the chance was that the user accepted a friendship request send by the socialbot (Figure 5). The triadic closure, interestingly, operated from the user side too, as the socialbots received a total of 331 friendship requests from their extended neighborhoods.

Second, the behavior depicted in Figure 4 matches the official statistics about users' login frequency in Facebook: 50% of the 750 million active Facebook users log on in any given day [1], and thus, it is expected that approximately half of the accepted

[12]Assuming that Facebook is mostly populated by genuine user profiles.

	Direct (%)		Extended (%)	
Profile Info	Before	After	Before	After
Gender	69.1	69.2	84.2	84.2
Birth Date	3.5	72.4	4.5	53.8
Married To	2.9	06.4	3.9	4.9
Worked At	2.8	4.0	2.8	3.2
School Name	10.8	19.7	12.0	20.4
Current City	25.4	42.9	27.8	41.6
Home City	26.5	46.2	29.2	45.2
Postal Address	0.9	19.0	0.7	1.3
Email Address	2.4	71.8	2.6	4.1
Phone Number	0.9	21.1	1.0	1.5
IM Account ID	0.6	10.9	0.5	0.8
Average	13.3	34.9	15.4	23.7

(a)



(b)

Figure 9: Data revelation before and after operating the SbN. The table in (a) shows the difference as a percentage of neighborhoods size, while the figure in (b) visualizes this difference in numbers. The direct neighborhoods consisted of 3,055 user profiles and the extended neighborhoods consisted of 1,085,785 user profiles (table data), adding up to 1,088,840 user profiles (figure data).

friendship requests are observed within one day of the requests being sent by the socialbots.

Third and last, the users who were infiltrated during the bootstrapping phase, that is, those who were selected at random, showed another expected correlation in their behavior [64]: the more friends they had, the higher the chance was that they accepted a friendship request from a socialbot (i.e., a stranger), as shown in Figures 6 and 7.

6.2. Harvested Data

As the socialbots infiltrated Facebook, they harvested a large set of user data. We were able to collect news feeds, user profile information, and "wall" messages—practically everything shared on Facebook by the infiltrated users—which could be used for large-scale user surveillance [18]. Even though adversarial surveillance, such as online profiling [79], is a serious online privacy concern, we decided to focus only on user data that have monetary value such as Personally Identifiable Information (PII).

After excluding all remaining friendships between the socialbots, the total size of all direct neighborhoods of the socialbots was 3,055 profiles. The total size of all extended neighborhoods, on the other hand, was as large as 1,085,785 profiles.

In Figure 9, we compare user data revelation of selected PII before and after operating the SbN.

6.3. *Infiltration Performance*

One way to judge whether the resulting infiltration was the making of a small number of "outstanding" socialbots is to compare them in terms of the number of user profiles they have infiltrated, as shown in Figure 8. Accordingly, we can group the socialbots into two leagues representing the two humps in the figure. The first one constitutes 86% of the socialbots and 70% of the overall infiltration, where each socialbot has infiltrated 0–50 user profiles. The second league, on the other hand, constitutes 10% of the socialbots and 23% of the overall infiltration, where each socailbot has infiltrated 60–80 user profiles. The rest of the socialbots constitute only 4% of the socialbots with 7% of the overall infiltration. Some of these socialbots, however, got blocked by the FIS.

After operating the SbN for eight weeks, only 20 of the user profiles controlled by the socialbots got blocked by the FIS, and curiously, all of them were controlled by f-socialbots. After further investigation, we found that these profiles were blocked because some Facebook users flagged them as spam.[13] In fact, we did not observe any evidence that the FIS detected what was really going on other than relying on user feedback, which seems to be an essential but potentially dangerous component of the FIS.

Finally, we noticed that employing the commands `cluster` and `status_update`, which are described in Table 2, had a desirable effect. It appears that the socialbots seemed more human-like as only 20% of the 102 profiles they controlled got blocked, as opposed to 93% of the 15 user profiles we used in our pilot study where we decided not to use these commands. This, in a sense, reflects one of the drawbacks of relying on user feedback.

7. Economic Analysis

In Section 5, we showed that OSNs, Facebook in particular, are vulnerable to large-scale infiltration campaigns run by an SbN, but is it economically feasible for a botherder to operate the SbN after all, or should he expect to lose money in order to sustain it? In what follows, we adopt the analysis of Herely [47] and reason about the economic feasibility of operating an SbN by a rational (i.e., profit-driven) botherder. As opposed to a botherder who is motivated by self-fulfillment, fun, or proof of skills, a *rational* botherder makes decisions about the SbN operation and its infiltration strategy based on whether the expected reward exceeds the cost (i.e., having a non-zero profit).

We aim to address questions related to: (1) the cost structure: how much does it cost to operate a particular SbN? (2) the infiltration scale: how many users to infiltrate in the targeted OSN? (3) the SbN size: how many socialbots to build and operate? (4) the infiltration strategy: what are the operational decisions that maximize the expected profit?

To tackle the questions above, we use a simple mathematical model that is useful for cost-volume-profit analysis [80]. In particular, we employ this model to analyze the scalability of infiltration in an economic context, and focus on the cost structure of an SbN as the infiltration grows arbitrarily large in scale, and its effect of profitability. We then devise two feasible infiltration strategies depending on the scalability requirements, along with an estimate of the SbN size. Finally, we apply the concepts we developed in practice and show that, for example, operating an SbN on Facebook is expected to be profitable but not particularly attractive as an independent and sustainable business.

7.1. *Mathematical Model*

The objective of the botherder is to maximize the profit by operating an SbN of size m using the best profit-maximizing infiltration strategy. Each socialbot has a fixed average cost $\bar{c}$ associated with it, which represents the cost of building a new socialbot. Moreover, every socialbot is limited to up to k social connections, where k is an OSN-specific parameter and is usually fixed but can be arbitrarily large. Every user profile in the targeted OSN has an associated average extracted value $\bar{v}$, which represents, for example, an estimate of the monetary value of the profile information. We denote the total cost and reward of infiltrating n user profiles by $C(n)$ and $R(n)$, respectively. Operating an SbN to achieve large-scale infiltration in a targeted OSN is profitable only if the reward exceeds the cost, that is, whenever the profit $P(n)$ is positive as follows:

$$P(n) = R(n) - C(n) > 0. \tag{3}$$

Finally, we refer to the value of n where the botherder breaks even by n_0, that is, $P(n_0) = 0$ where $n_0 > 0$.

7.2. *Scalability of Infiltration Campaigns*

We now formally define the scalability of an infiltration campaign from an economic perspective. Before we do so, however, let us present a brief, informal definition. We say that an SbN is scalable if the botherder has economic incentive to infiltrate ever more OSN users in a single campaign, that is, whenever $P(\alpha n) > \alpha P(n)$ for a scaling factor $\alpha > 1$. This means that operating a scalable SbN accrues an increasing marginal profit.

In the following sections, we construct the concepts needed to relate scalability to profitability. All of the functions we present are defined in terms of n, as we are interested in scaling the infiltration of an SbN rather than its size m. However, n and m can be related, which is an undesirable situation in terms of scalability, as we show in the following discussion.

7.2.1. *Scalability in Economic Context*

In this section, we generalize the scalability definition of online attacks presented by Herley [47], and adapt it to arbitrarily-large infiltration. Formally, we call an SbN *scalable* if $C(n)$

[13] Based on the content of a pop-up message that Facebook showed when we manually logged in using the blocked socialbots' profiles.

grows more slowly than $R(n)$ as $n \to \infty$, that is, $C(n) = \mathbf{o}(R(n))$, which is the case whenever the following equality holds:

$$\lim_{n\to\infty} \frac{C(n)}{R(n)} = 0, \tag{4}$$

and otherwise, we call the SbN *non-scalable*.

A scalable SbN is favorable as it has a desirable effect on profitability, where the following equality now holds:

$$\lim_{n\to\infty} \frac{P(n)}{R(n)} = 1, \tag{5}$$

which means that given a scalable SbN, $P(n)$ grows as fast as $R(n)$, or similarly, $P(n) \approx R(n)$ as $n \to \infty$, where $C(n)$ has a diminishing effect on profit.

The definition of scalability, as conditioned by Equation 4, is agnostic to the distribution of both $C(n)$ or $R(n)$, and it always holds as long as the condition $C(n) = \mathbf{o}(R(n))$ is satisfied. This, however, demands that we precisely define $C(n)$ and $R(n)$ for any given SbN in order to judge its scalability.

7.2.2. The Cost Structure

For an SbN of size m, let $C_s(n)$ and $C_o(n)$ be the setup and operation costs of the SbN, respectively, where n is the number of users to infiltrate in the targeted OSN. The *setup cost* is the cost incurred due to building the SbN components required for starting its operation, and the *operation cost* is the cost incurred due to running the SbN for a particular period of time. Accordingly, we define the total cost, $C(n)$, as follows:

$$C(n) = C_o(n) + C_s(n). \tag{6}$$

Given a rational botherder, the SbN is expected to be operated as part of an existing botnet, and thus, $C_o(n) = 0$ and $C(n) = C_s(n)$. Put differently, a rational botherder does not pay any fee for computational resources, but instead, he exploits "zombie" machines that are infected by his botnet's malware. In fact, this is the current practice of today's botnet owners. For example, the Koobface botnet consists of many components that are downloaded after the initial infection, including a key-logger, a CAPTCHA-solver, a DNS-changer, a data-stealer, an OSN propagation plug-in, and others [23]. Extending the discussion to include the *rental cost* of infected machines is straightforward, in which case $C_o(n)$ represents the rental cost. For example, recent report shows that an average price for renting a botnet is \$67 for 24 hours, and \$9 for hourly access [56]. In this article, we assume that the botherder controls both the botnet and the SbN, which is part of the threat model discussed in Section 4.2.

As outlined in Section 7.1, each socialbot is limited to up to k social connections. This limitation forces $C_s(n)$, which represents the cost of building $m = \lceil \frac{n}{k} \rceil$ socialbots, to become linearly dependent on n whenever $n > k$, as follows:

$$C_s(n) = C(n) = \begin{cases} \bar{c} & \text{if } n \le k, \\ \bar{c} \times \left\lceil \dfrac{n}{k} \right\rceil & \text{if } n > k. \end{cases} \tag{7}$$

From Equation 7 above, the total cost $C(n) = \mathbf{O}(1)$ for $n \le k$, where the botherder needs to build only a single socialbot. In this case, the following inequality holds for any scaling factor $\alpha > 1$, as long as $\alpha n \le k$:

$$C(\alpha n) < \alpha C(n), \tag{8}$$

and accordingly, we say that the SbN achieves *economy of scale*: the cost of operating an SbN decreases per infiltrated user profile as the number of users to infiltrate increases, and it takes effect after the initial setup cost has been redeemed. For $n > k$, however, $C(n) = \mathbf{O}(n)$ and the complement of Equation 8 holds for any scaling factor $\alpha > 1$, that is,

$$C(\alpha n) \ge \alpha C(n). \tag{9}$$

In this case, every additionally infiltrated user profile adds cost at least as much as the previous one, and the SbN consists of a growing number of socialbots m, which depends now on both n and k.

7.2.3. The Basic Reward Model

As there is no clear pricing structure for the SbN goods, we assume that $R(n)$ grows at least linearly as $n \to \infty$, that is, $R(n) = \mathbf{\Omega}(n)$. This assumption is usually made upon entering new markets that do not offer short-term feedback [80], in which case the simplest way for valuation is to consider the average extracted value $\bar{v}$ and quote it for each unit of goods. Such a simple linear reward model, however, does not accommodate the *network effect*, nor the market's pricing fluctuations or trends.[14] Thus, we believe this model is useful for basic, short-term analysis, which is suitable for today's underground economy, especially in the presence of dishonesty, cheating, and uncertainty [81].

To make things concrete, let $\bar{y}$ be the yield of the infiltration, which represents the average acceptance rate of a connection request sent by a socialbot. For now, we define $R(n)$ as follows:

$$R(n) = n \times \bar{y} \times \bar{v} = \mathbf{O}(n). \tag{10}$$

7.2.4. Scalability vs. Profitability

By considering the total cost and the expected reward of operating an SbN (Equations 7 and 10), our definition of scalability (Equation 4) leads us to the following basic result: for $n \le k$, the SbN is scalable when using a fixed number of socialbots (ideally, a single socialbot), but for $n > k$, the SbN becomes non-scalable when using a growing network of socialbots, whose size now depends on both n and k.

To give a meaningful interpretation of scalability, we now show the economic implication of the result shown above. Let us consider the profit when we scale the SbN operation by a

[14]The network effect or externality is the effect that one user of a good or service has on the value of that product to other people. When network effect is present, the value of a product or service is dependent on the number of others using it [80].

factor $\alpha > 1$. For $\alpha n \leq k$, we have:

$$\begin{aligned} P(\alpha n) &= R(\alpha n) - C(\alpha n) \\ &> \alpha R(n) - \alpha C(n) \\ &> \alpha P(n), \end{aligned} \tag{11}$$

which means that the botherder earns more profit by infiltrating, say, $100n$ users as opposed to n users when using a fixed number of socialbots. There is a clear economic incentive for him to use these socialbots to go at as large scale in infiltrating OSN users as possible. When $\alpha n > k$, however, the situation changes as follows:

$$\begin{aligned} P(\alpha n) &= R(\alpha n) - C(\alpha n) \\ &\leq \alpha R(n) - \alpha C(n) \\ &\leq \alpha P(n), \end{aligned} \tag{12}$$

that is, when operating an SbN consisting of a growing number of socialbots, there is no benefit, if not a loss, from scaling the SbN to infiltrate more OSN users. The botherder makes the same profit, at best, if he infiltrates n or $100n$ users. This results in the following interesting dilemma: the botherder has a high incentive to scale the infiltration using at least a single socialbot, but has not incentives to infiltrate even more users by operating a growing network of socialbots. Given the large-scale objective of any infiltration campaign, how can the botherder overcome the plight of non-scalability?

In the coming section, we show how a botherder can avoid this dilemma by deriving two types of profit-maximizing infiltration strategies.

7.3. Infiltration Strategies

As shown in the previous section, the botherder is faced with a scalability dilemma: it is rational for him to scale the infiltration using individual socialbots, but it is not as beneficial to do so if the SbN grows in size. What should the botherder do?

7.3.1. Scalable Data-driven Infiltration

One possible way to avoid this dilemma is to force the SbN to be scalable by reducing $C(n)$ from $\mathbf{O}(n)$ to $\mathbf{o}(n)$ whenever $n > k$. The botherder can do so by artificially removing the limit k on each socialbot, and thus, making the cost independent from n. One way to achieve this is to break one or more social connections between each socialbot and the OSN users it has infiltrated so far, whenever the limit k is reached and the connections have been already used for collecting user profile information. Even better, the botherder can order the socialbots to maintain at most $\bar{N}(G) = \eta$ social connections, and accordingly, fall into the average user profile category. In an SbN with m socialbots, doing so will result in the following new cost, $\hat{C}(n)$, which is now independent from n:

$$\hat{C}(n) = \bar{c} \times m = \mathbf{O}(1), \tag{13}$$

and thus, $\hat{C}(\alpha n) < \alpha \hat{C}(n)$ for a scaling factor $\alpha > 1$. The SbN is now scalable and achieves economy of scale—the botherder has clear incentives to scale its operation to infiltrate as many OSN users as possible.

Given that the botherder has to break social connections on the way, we call such an infiltration strategy *data-driven*: the botherder seeks to harvest as much data from OSN users as possible by following a simple infiltrate-collect-break strategy. In this strategy, the yield is refined to include the average data revelation ratio per user profile, $\bar{d}$, as follows:

$$\hat{y} = \bar{y} \times \bar{d}. \tag{14}$$

An interesting question now follows: what are the reasonable values of m and n that maximize the profit of a data-driven SbN? Let $f(m)$ be a penalty cost function that increases as m decreases. This function models the negative effect of having an isolated socialbot, which could result in a higher chance of being detected and blocked, as we discussed in Section 6.3. In this article, we decided to use a simple penalty function that models the cost of building new socialbots due to other bots being blocked, which could happen during a single run of the infiltration campaign.[15] Accordingly, we define $f(m)$ as follows:

$$f(m) = \bar{c} \times m \times \bar{p}, \tag{15}$$

where $\bar{p}$ is the probability of a socialbot to get blocked, given its collaboration with $m-1$ other socialbots. We assume that $f(m) \to 0$ as $m \to \hat{m}$, where $\hat{m}$ is the number of socialbots that are needed to avoid detection, that is, when $\bar{p} \approx 0$. Accordingly, we refine the $\hat{C}(n)$ to include the penalty cost function as follows:

$$\tilde{C}(n) = \hat{C}(n) + f(m) = \hat{C}(n) \times (1 + \bar{p}) = \mathbf{O}(1). \tag{16}$$

We now formulate the above question as the following profit-maximization problem:

$$\begin{aligned} \underset{n,m}{\text{maximize}} \quad & P(n,m) = \overbrace{n \times \hat{y} \times \bar{v}}^{R(n)} - \overbrace{\bar{c} \times m \times (1+\bar{p})}^{\tilde{C}(n)} \\ \text{subject to} \quad & m \geq 1 \text{ and } n \geq 1, \end{aligned}$$

In order to maximize the profit, the botherder needs to maximize the reward or minimize the cost. The optimal solution is to set $m = \hat{m}$ (i.e., when $\bar{p} \approx 0$) and set n as large as possible (i.e., there is no closed form solution for n). This means that as $n \to \infty$, the profit $P(n) \to \infty$ as well, which is an expected result given the scalability of this infiltration strategy. Empirically, we showed that setting $\hat{m} \approx \eta$ is a good start to significantly reduce the penalty $f(m)$, which is achieved by connecting each socialbot with all other socialbots before starting to infiltrate the OSN on a large scale (i.e., by executing the `cluster` command).

[15]The penalty cost function can also incorporate the risk of having one or few socialbots being blocked based on the observation that they break significantly large number of social connections, and thus, having more socialbots would reduce this risk to some extent.

7.3.2. *Non-Scalable Target-driven Infiltration*

Another strategy is not to break the social connections with the infiltrated users, but to infiltrate as many users as required in order to meet a contractually-agreed targeted infiltration size $\hat{n} \geq 1$, after which the botherder receives a lump sum payment ρ from a third-party. We call such a non-scalable strategy *target-driven*: the botherder operates the SbN on a limited scale so as to infiltrate exactly $\hat{n}$ users, which is informed by solving the following profit-maximization problem:

$$\begin{aligned} \underset{n,m}{\text{maximize}} \quad & P(n,m) = \rho - \overbrace{\bar{c} \times m \times (1 + \bar{p})}^{\tilde{C}(n)} \\ \text{subject to} \quad & m \geq \left\lceil \frac{n}{k} \right\rceil \text{ and } n \geq \hat{n} \end{aligned}$$

Similar to the discussion in the previous section, the optimal solution is to set the SbN size $m = \max\left\{\hat{m} \approx \eta, \lceil \frac{\hat{n}}{k} \rceil\right\}$, and to increase the value of n to as large as needed in order to infiltrate exactly $\hat{n}$ users with yield $\bar{y}$.

7.4. *The Case of Facebook*

At the time of writing, an active user on Facebook had an average of $\eta = 130$ friends and was limited to $k = 5,000$ friends [1]. We consider the scenario where a botherder operates an SbN as part of a botnet, and thus, the operation cost $C_o(n) = 0$ and the setup cost $C_s(n) = C(n)$. A botherder starts off by building $m = 130$ socialbots in order to reduce the chance of a socialbot being blocked with a fixed probability $\bar{p}$. Building a socialbot involves creating a new user profile, which often requires solving a CAPTCHA. We consider the situation where the botherder outsources the task of solving CAPTCHAs to one of the CAPTCHA-solving businesses, which have a service charge of as low as \$1 per 1,000 successfully-solved CAPTCHAs [54].[16] Accordingly, we set the average fixed cost of building a socialbot to $\bar{c} = \$0.001$, which means that the cost of building the SbN is $\hat{C}(n) = 0.001 \times 130 = \$0.13$ (Equation 13). Based on the results we showed in Section 6.3, we set $\bar{p} = 0.2$ and now the new total cost of the SbN is as small as $\tilde{C}(n) = 0.13 \times (1 + 0.2) = \$0.156$ (Equation 16). Notice that $\tilde{C}(n)$ represents the initial investment the botherder has to make in order to operate this SbN, and curiously, it is very small.

We start with an example of a data-driven SbN but consider a botherder who seeks to sell only email addresses.[17] According to the table in Figure 9(a), the average data revelation of email addresses is $\bar{d} = 0.718$. As the botherder is expected to operate the SbN mostly in the propagation phase, we set the average success rate of the infiltration to $\bar{y} = 0.59$, as shown in Section 5.4.3. This results in the new yield of $\hat{y} = 0.59 \times 0.718 = 0.4236$ (Equation 14). Based on a recent research by Symantec [82], 1MB of email addresses is worth \$0.3–\$40 in IRC underground markets when bought in bulk, with an average of \$20.15 per MB. Assuming that an email address is 15 ASCII characters long (i.e., 15 bytes), then an email address has a value of $\$3.0225 \times 10^{-4}$. Accordingly, we set the average extracted value of a user profile to this conservative value, that is, $\bar{v} = \$3.0225 \times 10^{-4}$. We can now calculate the profit using Equation 3 as follows:

$$\begin{aligned} P(n) &= R(n) - \tilde{C}(n) \\ &= n \times \hat{y} \times \bar{v} - \bar{c} \times m \times (1 + \bar{p}) \\ &= n \times 0.4236 \times 3.0225 \times 10^{-4} - 0.001 \times 130 \times (1 + 0.2), \end{aligned}$$

and the botherder breaks even when $n_0 = 1,219$ user profiles. To make \$1,000 in profit, the botherder has to infiltrate approximately 7.8 million users. In fact, the botherder earns \$96,025 by infiltrating the whole user population of Facebook, consisting of 750 million users [1]. We acknowledge, however, that targeting the whole Facebook population in a single campaign is unrealistic, but it is still useful to get an optimistic estimate of how much user data are worth in the underground market, even for the unlikely case when the botherder manages to run such a campaign without being detected.[18]

As for a target-driven SbN, the numbers look more promising for the botherder. Motoyama et al. [57] show that one can use online freelance markets to buy a thousand Facebook friends connections for \$26. The authors call such a task OSN *linking*, where a freelance worker takes on the job of establishing a particular number of social connections with users in a targeted OSN, after which the worker receives a lump sum payment. Accordingly, to receive a payment of $\rho = \$1,000$, the botherder is expected to infiltrated $\hat{n} = 38,462$ user profiles. Given the limit of $k = 5,000$ friends per socialbot, the botherder builds $m = \max\left\{130, \lceil \frac{38,462}{5,000} \rceil\right\} = 130$ socialbots, and is expected to make a profit of $P(n) = 1,000 - 0.001 \times 130 \times (1 + 0.2) = \$999.844$.

7.5. *The Big Picture*

So far, we showed that a large-scale infiltration campaign is both a real threat and profitable, but does it imply that such campaigns can be independent underground businesses as well? In what follows, we first discuss the implications of the underground markets on large-scale infiltration campaigns, and then argue that even though such a campaign is expected to make profit, it is still more reasonable to consider it as bootstrap campaign for later exploits, rather than an independent business.

7.5.1. *Underground Market Implications*

Using a scalable data-driven SbN has the implications that its operation ought to be automated and non-adaptive to individual OSN users, while delivering commodity-goods as a result. These implications are attributed to the economy of scale the strategy achieves, and are discussed in depth by Herely [47].

[16]CAPTCHA-solving businesses usually sell their service in bulk. For instance, the smallest unit of purchase could be 1,000 CAPTCHAs. We assume, however, that it is possible to buy smaller units of service.

[17]We chose to restrict the harvested data to email addresses in order not to overestimate the extracted value of user profile information.

[18]The same IRC study [82] estimates a minimum of $\bar{v} = \$0.9$ per *full identity*, which resembles a full Facebook profile information. Accordingly, using an average profile data revelation of $\bar{d} = 0.349$ (table in Figure 9), an optimistic estimation is to expect the botherder to make \$185,319 in profit by infiltrating one million Facebook users.

In particular, adding per-user personalization, such as responding to individual messages requesting introductions, or adapting to per-user countermeasures, such as breaking CAPTCHAs to send more connection requests, violates the cost structure of the strategy and would result in potential losses. Moreover, as the operation of the SbN is fully automated, other copycat attackers will have the incentives to build and operate similar SbNs as well. As Herley puts it [47]: "once a clever exploit is scripted, then it can be used by many." Consequently, this automation will highly increase the demand for large-scale infiltration but decrease the associated profit, resembling the effect of the *tragedy of commons*: a dilemma arising from the situation in which multiple individuals, acting independently and rationally, will ultimately deplete a shared limited resource, even when it is clear that it is not in anyone's long-term interest [83]. This has the implication of decreasing $\bar{v}$ down to zero.

In a non-scalable target-driven SbN, the implications are different. As the objective of the botherder is to reach a targeted infiltration size rather than to harvest user profile information, it makes more sense to adapt the SbN to first target OSN users who increase the yield. As shown in Section 6.1, spending extra effort in finding user profiles who have higher than average number of social connections is expected to achieve this objective. Interestingly, the distribution of OSN users according to the number of social connections they have is usually a power-law [59], which has a considerable concentration that underlines the 80/20 rule (e.g., Zipf, Pareto). This means that the botherder will target a relatively small number of OSN users who have a significantly large number of social connections. Accordingly, this allows the botherder to undertake both infiltration strategies at the same time, without having to worry about conflicting or competing objectives.

7.5.2. Large-scale Infiltration as a Business

Even though the profit estimates in Section 7.4 might be optimistic, it is still safe to assume that operating an SbN in a targeted OSN is expected to make a non-zero profit, but would a rational botherder be interested in running such a business? In what follows, we argue that large-scale infiltration is not a particularly attractive business, and a rational botherder would utilize his SbN for subsequent, more profitable campaigns.

First, notice that the analysis we provide is a cost-volume-profit analysis, which is useful for short-run decisions and has to be reevaluated regularly. Even though the botherder will be able to observe market trends over time, which allows him to plan a pricing structure, use more predictive economic models, and forecast future profits, the underground market economy is still unfriendly. Herley and Florêncio [81] use simple economic arguments to show that such markets suffer from cheating, dishonesty, and uncertainty. Contrary to what is generally assumed, they show that these markets represent a classic example of *markets for lemons* [84]: the situation where sellers have better information than buyers about the quality of their goods, which leads to an adverse selection where the bad drives out the good.[19] Ultimately, these markets drive "high-quality" businesses out of existence and result in a diminishing valuation of the markets' goods. After all, "nobody sells gold for the price of silver." [81]

Second, maintaining a business requires durability. Operating an SbN, on the other hand, is expected to be more costly to maintain by time, as the OSN security defenses will be updated to mitigate its infiltration, especially when an adversarial learning system is deployed. This will result in an arms race between the botherder and the targeted OSN, where the more resourceful party will eventually win. Based only on the profit the botherder is expected to make out of operating an SbN, the odds are small that his investment in maintaining and updating the SbN will payback, especially because this would force his SbN to be non-scalable, as discussed in Section 7.5.1.

Third and last, Kanich et al. [85] show that the underground market of spam-advertised businesses is particularly attractive, where such businesses make hundreds of thousands of dollars a month in revenue. The botherder is thus better off using an SbN as a tool to run a subsequent, large-scale, and personalized e-mail spam campaign as part of a larger underground affiliation.

8. Implications For Other Systems

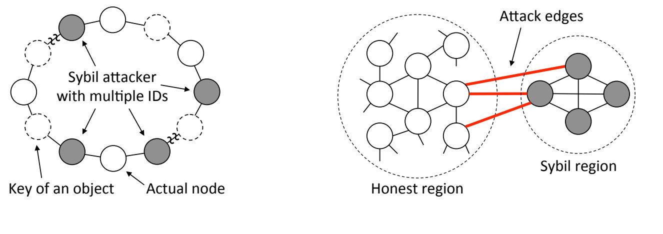
So far, we showed that running a large-scale infiltration campaign is feasible in practice and a has a low cost associated with it. This section explores the wider implications of large-scale infiltration in OSNs beyond today's social Web. In particular, we show that large-scale infiltration has alarming implications on software systems that use the social graph of OSNs to personalize, fine-tune, or bootstrap socially-aware services. We first outline the common assumption these systems make about the capabilities of adversaries in OSNs, and then, we focus on a case study in order to show that this assumption is generally unsafe and can lead to undesirable situations.

8.1. The Capability of a Social Adversary

Today, many software systems are socially-aware due to the availability of huge and centralized OSNs that offer easy-to-use APIs. This enabled the development of a new set of socially-aware services that rely on the social graph of the used OSN. For example, OSNs are used to defend against Sybil attacks in peer-to-peer systems [14], enable knowledge-sharing and personalization in social search [86], model trust in recommender systems [87], and cooperate safely in online peer-based backup systems [88]. All of these systems, however, make the following assumption implicitly or explicitly about the capabilities of an adversary in OSNs:

> *It is particularly difficult for an adversary to establish arbitrarily many social connections between his OSN accounts (i.e., Sybils) and the rest of the users in the social graph.*

[19] A lemon is an American slang term for a car that is found to be defective only after it has been bought. The market for lemons concludes that owners of good cars will not place their cars on the used-car market. This is sometimes summarized as "the bad driving out the good" in the market.



(a) Distributed Hash Table (b) Social Network

Figure 10: Detecting Sybil nodes in a typical DHT. In (a), dashed nodes represent keys of objects drawn from an identifier space, and un-dashed nodes represent peers where each peer has a unique identifier drawn from the same identifier space. In (b), an edge is added between two nodes if these nodes trust each other, forming a network of trust among the peers in the DHT. The structure of the resulting social network can be used to label each node either Sybil or honest (i.e., non-Sybil). This allows the classification of nodes in the DHT into two regions: the Sybil region and the honest region.

This assumption has the indirect effect of transforming the used OSN into a trust network, which brings in many advantages such as a well-defined trust metric, credibility, and hassle-free integration. In the following section, we evaluate how realistic this assumption is using a case study, and show that the outcomes of the studied system can drastically change once the assumption above does not hold in practice.

8.2. Case Study: Sybil Defenses via Social Networks

As outlined in Section 2.2, the Sybil attack refers to the situation where an adversary controls a set of fake identities, each called a Sybil, and joins a targeted system multiple times under these Sybil identities.[20] The adversary can then mount many follow-up attacks using these Sybil identities in order to disrupt the targeted system. For example, an adversary can pollute the voting scheme of a trust and reputation system [90], subvert the routing and data replication services in a Distributed Hash Table (DHT) [91], or cripple many critical functions of a wireless sensor network such as routing, resource allocation, an misbehavior detection [92]. In this section, we focus on Sybil attacks in DHTs: decentralized, distributed systems that provide a lookup service, similar to a hash table, for a group of collaborating peers or nodes.

To defend against the Sybil attack, recent research propose to exploit the implicit social network among the nodes of a DHT in order to detect, and thereof limit, the number of Sybil nodes it may contain [93–97]. To demonstrate the main idea behind these Sybil defenses, let us consider the DHT in Figure 10(a). Each node is assigned a unique identifier and is responsible for maintaining a set of (key, value) pairs of shared objects, which are stored and maintained in the DHT. Any participating node can efficiently retrieve the value associated with a given key of an object by following the DHT protocol (e.g., Chord [98], Kademlia [99]). Figure 10(a) illustrate the case where an adversary joins the DHT under four Sybil identities, which are represented as gray nodes.

A social network-based *Sybil defense protocol* is used to first construct the trust network among the peers in the DHT, and then use the graph structure of this social network to detect the Sybil nodes in the DHT. In particular, the protocol classifies the peers in the social network into two groups called *regions*: the Sybil region consisting of mostly Sybil nodes in the DHT, and the *honest* region consisting of mostly non-Sybil nodes, as shown in Figure 10(b). The edges in the social network connecting the Sybil region to the honest region are called the *attack edges*, and their quantity plays an important role in the accuracy of the classification. The basic idea is as follows: even if an adversary controls many Sybil nodes in the DHT, it is assumed that it is particularly difficult for him to establish many trust relationships with honest nodes in the social network, that is, it is hard to increase the number of attack edges in the network. Thus, one can devise a detection mechanism that relies on this observation in order to group the nodes into honest and Sybil regions, and hence, cluster the network.

Most of the Sybil defenses via social networks use techniques that are based on random walks and mixing times in a given social graph [93–97]. Yu [14], however, shows that if an adversary introduces few Sybil nodes such that the mixing time of the social graph is not affected, then no approach based on the mixing time can possibly tell that these nodes are Sybil. The *mixing time* describes how fast random walks in a given graph approach the stationary distribution of that graph, and is not affected as long as the adversary introduces at most as many Sybil nodes as the number of attack edges. Thus, if the mixing time is not effected by the introduction of Sybils, none of the major Sybil defenses will work effectively.[21] This result is not surprising as one would expect that if an adversary manages to establish many trust relationships with honest nodes, then the Sybil nodes will integrate into the honest region, rendering ineffective any clustering technique that is solely based on the social graph.

The above conclusion extends to all open-access systems that integrate OSNs such as Facebook, especially those that are Web-based. A system is called *open-access* if it allows any user to join the system by providing an identity that is issued by the system itself or by other third-party identity providers (e.g., Facebook Single Sign-On [60], OpenID [100]). For example, Facebook reports that more than 500 million of its users interact with third-party applications on its Web platform or experience Facebook platform on other websites every month, where more than seven million applications and websites are integrated with Facebook [1]. Apparently, this is a welcoming haven for Sybil attackers in today's social Web, including SbN botherders and their affiliates.

9. Challenges and Countermeasures

Defending against large-scale infiltration in OSNs can be divided into prevention and limitation. To prevent an SbN from

[20]The attack is named after the subject of the book Sybil [89], a case study of a woman with multiple personality disorder.

[21]The interested reader can refer to [14] for a formal treatment of this topic.

operating in an OSN, the OSN operator has to eliminate the factors that enable the problem in the first place, that is, to fix at least one of the vulnerabilities outlined in Section 3. Doing so, however, gives rise to a set of socio-technical challenges that relate to web automation, online-offline identity binding and usable security. Limiting large-scale infiltration, on the other hand, implies that the OSN operator accepts the fact that such an infiltration is possible or existent, and consequently, uses techniques that aim to limit the infiltration and its implications.

In this section, we first present the key challenges in defending against malicious socialbots. After that, we discuss possible countermeasures that are used to detect and limit the operation of large-scale adversarial campaigns in OSNs.

9.1. Challenges

Our objective here is *not* to reduce the severity of the problem. Instead, we aim to eliminate its enabling factors, that is, to fix one or more of the vulnerabilities found in today's OSNs.

9.1.1. Web Automation

To simulate a user browsing an OSN, the adversary can employ *web automation* techniques, which include methods for solving CAPTCHAs, creating and populating multiple OSN accounts, crawling the social graph, and executing online social activities. Preventing this automation, however, requires solving at least one of the following challenges.

Challenge 1. *Design a reverse Turing test that is usable and effective even against "illegitimate" human solvers.*

A *reverse Turing test*, such as CAPTCHA [50], is a test that is administered by a machine and is designed to tell humans and machines apart. A *perfect* test presents a problem that is easy enough for all humans to solve, but is still impossible for a machine or an automation software to pass. Unfortunately, even a perfect test is ineffective if humans are exploited to solve the test in an *illegitimate setting*: the situation where human users are employed or tricked into solving reverse Turing tests that are not addressed to them. Under this setting, we refer to such a human user as *illegitimate*.

Eliminating the economic incentives for underground businesses that employ illegitimate human solvers is a first step towards tackling this challenge [54], but it does not solve it as legitimate users can still be tricked and situated into illegitimate settings, which is the case for the Koobface botnet [23]. This demands the design of new reverse Turing tests that are resilient to even those illegitimate users, which we believe is generally difficult to achieve.

Fast-response CAPTCHAs, for example, require the test to be solved in a relatively shorter time, as opposed to typical implementations. This makes it more difficult for automation scripts to pass the test, as they require extra time to relay the test, solve it and respond back. Fast-response CAPTCHAs, however, are expected to put more pressure on legitimate users who require easy and fast access to online services, and could potentially repel them away from using them.

Alternatively, authenticating users via their social knowledge (e.g., whether they can identify friends from photos), can be used as an effective test that is challenging for illegitimate users to solve [101]. Other that its usability issues, Kim et al. [102] show that it is relatively easy to circumvent such a social authentication scheme by either guessing, automated face recognition, or social engineering.

Challenge 2. *Effectively limit large-scale Sybil crawls of OSNs without restricting users' social experience.*

A *large-scale crawl* is a malicious activity where an adversary manages to crawl large portions of a target OSN, including both the social graph and all accessible users' profile information. Today, large-scale crawls are mitigated by employing a network-wide audit service, which limits the number of profiles a user can view per account or IP address in a given period of time [31]. This, however, can be circumvented by using a set of accounts, each called a *Sybil*, and then performing *Sybil crawling* on a large scale, typically using a botnet with multiple IP addresses [23].

To overcome this drawback, one can use the knowledge about the social graph to effectively limit Sybil crawls. Genie [103], for example, is a system that models the trust between users in an OSN as a *credit network*, where a user can view the profile of another user only if the path between them in the social graph has enough credits to satisfy the operation. If an adversary who controls many Sybil accounts attempts to crawl the OSN on a large scale, then Genie guarantees that the adversary will exhaust all the credits on the paths connecting the Sybil accounts to the rest of the network, thus limiting large-scale Sybil crawls. This approach, however, is based on the assumption that it would be hard for an adversary to establish an arbitrarily large number of social relationships with other users, which we showed to be an unsafe assumption, especially in OSNs as Facebook.

Challenge 3. *Detect abusive and automated usage of OSN platforms and social APIs across the Internet.*

In concept, *malicious automation* represents the situation where an adversary scripts her way of consuming system's resources in order to cause damage or harm to the system itself or its users. *Abusive automation*, on the other hand, is less severe where the adversary exploits the offered service in violation of the declared Terms of Service. From the OSN operator standpoint, all HTTP requests come from either a browser or through the social API, which is intentionally provided to support automation. Requests that are not associated with a browsing session, that is, those that do not append the required session cookies, can be easily detected and dealt with. With web automation, however, an adversary can simulate an OSN user and make all requests look as if they originate from a browser. Moreover, the patterns at which these requests are made can be engineered in such a way that makes them fall under the normal traffic category. In order to uncover adversarial campaigns, it is important to reliably identify whether such requests come from a human or a bot, along with means to distinguish patterns of abusive activities, even if the adversary has a knowledge of the used classification techniques.

Looking for regularities in the times at which requests are made, for example, can be used to detect automation in OSNs [104]. This, however, can be easily circumvented by simply mimicking the times and irregularities at which a human user makes such requests.

9.1.2. Identity Binding

Most of the challenges we presented so far are difficult due to the capability of the adversary to mount the Sybil attack. This leads us to the following challenge:

Challenge 4. *Guarantee an anonymous, yet credible, online-offline identity binding in open-access systems.*

Douceur [39] shows that without a centralized trusted party that certifies online identities, Sybil attacks are always possible except under extreme and unrealistic assumptions of resource parity and coordination among participating entities. Thus, limiting the number of Sybil accounts by forcing a clean mapping between online and offline identities is widely recognized as a hard problem, especially given the scalability requirements of today's online, open-access software systems.

Arguably, one way to tackle this challenge is to rely on governments for online identity management just as in offline settings. The *open government* initiative [105], for example, enables U.S. citizens to easily and safely engage with U.S. government websites using open identity technologies such as OpenID. This, however, requires creating *open trust frameworks* [105] that enable these websites to accept identity credentials from third-party identity providers, a task that involves solving challenging issues related to identity anonymity, scalability, security, technology incentives and adoption [106, 107].

9.1.3. Usable Security

As part of computer security, *usable security* aims to provide the users with security controls they can understand and privacy they can control [108]. In OSNs such as Facebook, there appears to be a growing gap between what the user expects from a privacy control and what this control does in reality [49]. Even if the most sophisticated OSN security defense is in place, an OSN is still vulnerable to many threats, such as social phishing [24], in case its users find it puzzling to make basic online security or privacy decisions. This gives us strong motives to study the human aspect of the OSN security chain, which is by itself a challenge.

Challenge 5. *Develop OSN security and privacy controls that help users make informed decisions.*

Designing security controls that better communicate the risks of befriending a stranger, for example, might be practically effective against automated social engineering. This, however, requires eliciting and analyzing the befriending behavior of users, including the factors that influence their befriending decisions, in order to inform a user-centered design for such controls.

9.2. Countermeasures

The main goal of today's OSN security defenses is to detect, and thereby limit the operation or the potential impact of large-scale adversarial campaigns. We divide the following discussion into two parts based on the used defense approach, and briefly survey related work on the topic.

9.2.1. Social Network Analysis Approach

In large-scale infiltration, the operation of an SbN is particularly susceptible to detection during the bootstrapping phase, as the socialbots try to infiltrate users picked at random in the targeted OSN. Specifically, the socialbots in this phase are expected to have a significantly different neighborhood structure, as opposed the structure of the neighborhoods of genuine OSN users who belong to a community of friends.

For example, let us consider the *clustering coefficient*: a measure of how probable it is that two friends of a user are friends themselves, which is used to describe the degree to which nodes in a graph tend to cluster together [109]. Accordingly, one would expect the clustering coefficient to be relatively large when considering the neighborhood of a genuine Facebook user, but significantly smaller for the neighborhood of a socialbot during the bootstrapping phase. Shrivastava et al. [33] show that detecting OSN users that exhibit such abnormality in their clustering coefficients is an NP-complete problem, and therefore, they propose heuristics that mine the social graph to find such users. Even if such heuristics were efficient in a social network consisting of millions of users, the infiltration phase of the SbN has the effect of increasing the clustering coefficient of the neighborhood of each socialbot. This is the case because each socialbot will start to infiltrate the users in its extended neighborhoods, and gradually form clusters of friends.

Another technique is to exploit the *interaction graph* of an OSN: a modified version of the social graph, where an edge between two users in the original graph is removed if the recorded social interaction between these two users is below a predefined threshold or rate. Using such a graph, one can employ techniques similar to Sybil defenses via social networks (as desribed in Section 8.2), where now the interaction graph is used to detect Sybil nodes in the social graph based on the assumption that it is difficult for socialbots to establish a long, two-way interaction with genuine OSN users. Wilson et al. [110], however, show that using an interaction graph to detect Sybil nodes is not effective as the interaction graph has different structural properties that cause these techniques to perform poorly.

9.2.2. Machine Learning Approach

One can make the following observation about the operation of an SbN: it is expected that each socialbot have an alarmingly large number of friendship requests that have been rejected, and thus, a *binary classifier* can be used to classify user accounts into malicious or benign accounts based on the rejection rate. Yang et al. [30] show that Sybil accounts in Renren[22], one of China's biggest OSNs, exhibit such distinguishable behaviors.

[22] `http://www.renren.com`

They show that (1) friendship requests sent by Sybil accounts have a relatively high rejection rate, and (2) such Sybil user accounts tend to accept most of the requests they receive from other OSN users, send a relatively large number of friendship requests, and have a relatively small clustering coefficient.

The botherder, however, has clear incentives to mutate his SbN operation in order to circumvent classifiers that employ such features, as follows:

First, by clustering the socialbots in the bootstrapping phase, the botherder can avoid setting off alarms based on the initial η friendship requests per socialbot, as they will be automatically accepted by the other bots. Doing so might reduce the chance of a learning system to detect these bots, but the subsequent requests that are sent during the bootstrapping phase are still expected to have high rejection rate.

Second, a resourceful botherder can employ adversarial classifier reverse engineering techniques [77] in order to learn sufficient information about the deployed classifiers in the targeted OSN, and then construct an adversarial attack that is hard to detect. This, however, is part of the adversarial machine learning life-cycle, where learning systems, such as the Facebook Immune System (FIS) [31], are designed to learn from and adapt to successful cyber attacks, especially those attacks which are new and have not been seen before. Thus, over time, the most resourceful party will win. In terms of our experiment on Faceobok, we believe that the FIS has now valuable and detailed data about the SbN operation, which can be used to train its classifiers in order to detect similar campaigns in the future.

10. Conclusion and Future Work

> "If you know the enemy and know yourself, you need not fear the result of a hundred battles. If you know yourself but not the enemy, for every victory gained you will also suffer a defeat. If you know neither the enemy nor yourself, you will succumb in every battle."
>
> — Sun Tzu, The Art of War

From a computer security perspective, the concept of socialbots is both interesting and disturbing: the threat is no longer from a human controlling or monitoring a computer, but from exactly the opposite.

In this article, we evaluated how vulnerable OSNs are to a large-scale infiltration by a Socialbot Network (SbN). We used Facebook as a representative OSN, and found that using bots that mimic real OSN users is effective in infiltrating Facebook on a large scale, especially when the users and the bots share mutual friends. Moreover, such socialbots make it difficult for OSN security defenses, such as the Facebook Immune System, to detect or stop an SbN as it operates. Unfortunately, this has resulted in alarming privacy breaches and serious implications on other socially-aware software systems. In addition, we showed that a profit-driven adversary would not consider an SbN as an independent business, but would use it as a tool for subsequent and more profitable adversarial campaigns.

As with other online attacks, defending against malicious socialbots is an arms race where the objective of the defender is to limit any potential harm or damage, that is, to extend the time at which the system maintains its safe state. In order to effectively defend against malicious socialbots, one has to either limit their implications or prevent their operation all together. The latter case, however, involves fixing a set of inherent vulnerabilities found in today's OSNs, which boils down to solving a number of socio-technical challenges that relate to web automation, online-offline identity binding and usable security.

We believe that large-scale infiltration in OSNs is only one of the many emerging cyber threats, and defending against such threats is the first step towards maintaining a safer social Web for millions of active web users.

To this end, we are currently investigating two directions from the defense side. The first involves understanding the factors that influence user decisions on befriending strangers, which is useful in designing user-centered security controls that better communicate the risks of online threats. The second, on the other hand, involves characterizing OSN-based Sybil defenses under the assumption of a social adversary, which could provide us with insights on how to design similar defenses that are resilient to such a resourceful adversary.

11. Acknowledgments

We would like to thank Shang Cai, San-Tsai Sun, Elizeu Santos-Neto, Albina Muslukhova, and Bader AlAhmad for their kind help and advice. We also would like to thank Laks V.S. Lakshmanan, Cormac Herley, Miranda Mowbray, Hasan Cavusoglu, and Adriana Iamnitchi for their feedback on an early draft of this article. This research is partially supported through funding from the NSERC Internetworked Systems Security Network (ISSNet) and GRAND NCE. The first author is thankful to the University of British Columbia for a generous doctoral fellowship.

References

[1] Facebook, Facebook Statistics, `http://www.facebook.com/press` (March 2011). 1, 3, 7, 10, 14, 16

[2] Twitter, Twitter Numbers, `http://blog.twitter.com/2011/03/numbers.html` (March 2011). 1

[3] A. M. Kaplan, M. Haenlein, Users of the world, unite! the challenges and opportunities of social media, Business Horizons 53 (1) (2010) 59 – 68. 1

[4] G. Livingston, Social media: The new battleground for politics, `http://mashable.com/2010/09/23/congress-battle-social-media/` (September 2010). 1

[5] J. A. Vargas, Obama raised half a billion online, `http://voices.washingtonpost.com/44/2008/11/obama-raised-half-a-billion-on.html` (November 2008). 1

[6] F. Salem, R. Mourtada, Civil movements: The impact of Facebook and Twitter, The Arab Social Media Report 1 (2). 1

[7] C. Taylor, Why not call it a Facebook revolution?, `http://edition.cnn.com/2011/TECH/social.media/02/24/facebook.revolution/` (February 2011). 1

[8] D. Boyd, Social media is here to stay... Now what?, *Microsoft Research Tech Fest* (February 2009). 1

[9] Y. Boshmaf, I. Muslukhov, K. Beznosov, M. Ripeanu, The socialbot network: when bots socialize for fame and money, in: Proceedings of the 27th Annual Computer Security Applications Conference, ACSAC '11, ACM, New York, NY, USA, 2011, pp. 93–102. 1, 2

[10] Y. Boshmaf, I. Muslukhov, K. Beznosov, M. Ripeanu, Key challenges in defending against malicious socialbots, in: Proceedings of the 5th USENIX conference on Large-scale exploits and emergent threats, LEET'12, USENIX Association, Berkeley, CA, USA, 2012. 1, 2

[11] T. Fong, I. Nourbakhsh, K. Dautenhahn, A survey of socially interactive robots, Robotics and Autonomous Systems 42 (2003) 143–166. 1

[12] Z. Coburn, G. Marra, Realboy: Believable twitter bots, `http://ca.olin.edu/2008/realboy` (April 2011).
URL `http://ca.olin.edu/2008/realboy`

[13] T. Hwang, I. Pearce, M. Nanis, Socialbots: voices from the fronts, interactions 19 (2) (2012) 38–45. `doi:10.1145/2090150.2090161`.
URL `http://doi.acm.org/10.1145/2090150.2090161` 1, 2, 3, 4

[14] H. Yu, Sybil defenses via social networks: a tutorial and survey, SIGACT News 42 (2011) 80–101. 1, 2, 15, 16

[15] Q. Cao, M. Sirivianos, X. Yang, T. Pregueiro, Aiding the detection of fake accounts in large scale social online services, in: Proc. NSDI, 2012. 1, 2

[16] E. Morozov, Swine flu: Twitter's power to misinform, `http://neteffect.foreignpolicy.com/posts/2009/04/25/swine_flu_twitters_power_to_misinform` (April 2009). 1

[17] K. Thomas, C. Grier, V. Paxson, Adapting social spam infrastructure for political censorship. 1

[18] J. Nolan, M. Levesque, Hacking human: data-archaeology and surveillance in social networks, SIGGROUP Bull. 25 (2) (2005) 33–37. 1, 3, 10

[19] J. Bollen, H. Mao, X. Zeng, Twitter mood predicts the stock market, Journal of Computational Science 2 (1) (2011) 1 – 8. 1

[20] J. Bates, Sniffing out socialbots: The combustive potential of social media-based algorithms, `http://www.businessinsider.com/sniffing-out-socialbots-the-combustive-potential-of-social-media-based-algorithms-2011-12` (December 2011). 1

[21] J. Ratkiewicz, M. Conover, M. Meiss, B. Gonçalves, S. Patil, A. Flammini, F. Menczer, Truthy: mapping the spread of astroturf in microblog streams, in: Proceedings of the 20th international conference companion on World wide web, WWW '11, ACM, New York, NY, USA, 2011, pp. 249–252. 1, 3

[22] G. Yan, G. Chen, S. Eidenbenz, N. Li, Malware propagation in online social networks: nature, dynamics, and defense implications, in: Proceedings of the 6th ACM Symposium on Information, Computer and Communications Security, ASIACCS '11, ACM, New York, NY, USA, 2011, pp. 196–206. 2

[23] J. Baltazar, J. Costoya, R. Flores, The real face of Koobface: The largest web 2.0 botnet explained, *Trend Micro Research* (July 2009). 2, 3, 4, 6, 12, 17

[24] T. N. Jagatic, N. A. Johnson, M. Jakobsson, F. Menczer, Social phishing, Commun. ACM 50 (10) (2007) 94–100. 2, 3, 18

[25] S. Patil, Social network attacks surge, `http://www.symantec.com/connect/blogs/social-network-attacks-surge` (June 2011). 2

[26] Jet bots, `http://allbots.info` (2011). 2

[27] A. Pitsillidis, K. Levchenko, C. Kreibich, C. Kanich, G. M. Voelker, V. Paxson, N. Weaver, S. Savage, Botnet judo: Fighting spam with itself, in: NDSS, 2010. 2

[28] C. Grier, K. Thomas, V. Paxson, M. Zhang, @spam: the underground on 140 characters or less, in: Proceedings of the 17th ACM Conference on Computer and Communications Security, CCS '10, ACM, New York, NY, USA, 2010, pp. 27–37. `doi:http://doi.acm.org/10.1145/1866307.1866311`.
URL `http://doi.acm.org/10.1145/1866307.1866311`

[29] G. Stringhini, C. Kruegel, G. Vigna, Detecting spammers on social networks, in: Proceedings of the 26th Annual Computer Security Applications Conference, ACSAC '10, ACM, New York, NY, USA, 2010, pp. 1–9. `doi:http://doi.acm.org/10.1145/1920261.1920263`.
URL `http://doi.acm.org/10.1145/1920261.1920263`

[30] Z. Yang, C. Wilson, X. Wang, T. Gao, B. Y. Zhao, Y. Dai, Uncovering social network sybils in the wild, in: IMC, Berlin, Germany, 2011. 2, 7, 18

[31] T. Stein, E. Chen, K. Mangla, Facebook immune system, in: Proceedings of the 4th Workshop on Social Network Systems, SNS '11, ACM, New York, NY, USA, 2011, pp. 8:1–8:8. `doi:http://doi.acm.org/10.1145/1989656.1989664`.
URL `http://doi.acm.org/10.1145/1989656.1989664` 2, 3, 7, 17, 19

[32] C. Wagner, S. Mitter, C. Körner, M. Strohmaier, When social bots attack: Modeling susceptibility of users in online social networks, WWW'12 Workshops: Making Sense of Microposts (# MSM2012) (2012) 2. 2, 3

[33] N. Shrivastava, A. Majumder, R. Rastogi, Mining (social) network graphs to detect random link attacks, in: Data Engineering, 2008. ICDE 2008. IEEE 24th International Conference on, 2008, pp. 486 –495. `doi:10.1109/ICDE.2008.4497457`. 2, 18

[34] B. Viswanath, A. Post, K. P. Gummadi, A. Mislove, An analysis of social network-based sybil defenses, in: Proceedings of the ACM SIGCOMM 2010 conference on SIGCOMM, SIGCOMM '10, ACM, New York, NY, USA, 2010, pp. 363–374. 2

[35] S. Fortunato, Community detection in graphs, Physics Reports 486 (3-5) (2010) 75 – 174. `doi:DOI:10.1016/j.physrep.2009.11.002`.
URL `http://www.sciencedirect.com/science/article/pii/S0370157309002841` 2

[36] A. Rapoport, Spread of information through a population with socio-structural bias: I. assumption of transitivity, Bulletin of Mathematical Biology 15 (1953) 523–533, 10.1007/BF02476440.
URL `http://dx.doi.org/10.1007/BF02476440` 2, 7

[37] M. Nanis, I. Pearce, T. Hwang, Pacific social architecting corporation: Field test report, `http://pacsocial.com/` (November 2011). 3, 4

[38] M. Mowbray, The twittering machine, in: Proceedings of the 6th International Conference on Web Information Systems and Technologies, ASIACCS '11, 2010, pp. 299–304. 3

[39] J. R. Douceur, The sybil attack, in: IPTPS '01: Revised Papers from the First International Workshop on Peer-to-Peer Systems, Springer-Verlag, London, UK, 2002, pp. 251–260. 3, 18

[40] F. Nagle, L. Singh, Can friends be trusted? exploring privacy in online social networks, in: Proceedings of the 2009 International Conference on Advances in Social Network Analysis and Mining, IEEE Computer Society, Washington, DC, USA, 2009, pp. 312–315. `doi:10.1109/ASONAM.2009.61`.
URL `http://portal.acm.org/citation.cfm?id=1602240.1602706` 3, 7

[41] R. Potharaju, B. Carbunar, C. Nita-Rotaru, ifriendu: leveraging 3-cliques to enhance infiltration attacks in online social networks, in: Proceedings of the 17th ACM conference on Computer and communications security, CCS '10, ACM, New York, NY, USA, 2010, pp. 723–725. `doi:10.1145/1866307.1866410`.
URL `http://doi.acm.org/10.1145/1866307.1866410`

[42] D. Irani, M. Balduzzi, D. Balzarotti, E. Kirda, C. Pu, Reverse social engineering attacks in online social networks, Detection of Intrusions and Malware, and Vulnerability Assessment (2011) 55–74. 3

[43] L. Bilge, T. Strufe, D. Balzarotti, E. Kirda, All your contacts are belong to us: automated identity theft attacks on social networks, in: WWW '09: Proceedings of the 18th International Conference on World Wide Web, ACM, New York, NY, USA, 2009, pp. 551–560. `doi:http://doi.acm.org/10.1145/1526709.1526784`. 3, 4

[44] R. J. Anderson, Security Engineering: A Guide to Building Dependable Distributed Systems, 2nd Edition, Wiley Publishing, 2008. 3

[45] G. Brown, T. Howe, M. Ihbe, A. Prakash, K. Borders, Social networks and context-aware spam, in: CSCW '08: Proceedings of the ACM 2008 Conference on Computer Supported Cooperative Work, ACM, New York, NY, USA, 2008, pp. 403–412. `doi:http://doi.acm.org/10.1145/1460563.1460628`. 3

[46] M. Huber, S. Kowalski, M. Nohlberg, S. Tjoa, Towards automating social engineering using social networking sites, Computational Science and Engineering, IEEE International Conference on 3 (2009) 117–124. `doi:http://doi.ieeecomputersociety.org/10.1109/CSE.2009.205`. 3, 6

[47] C. Herley, The plight of the targeted attacker in a world of scale, in: The 9th Workshop on the Economics of Information Security (WEIS 2010), 2010. 4, 11, 14, 15

[48] The Web Ecology Project, The 2011 socialbots competition, `http://www.webecologyproject.org/category/competition/` (January 2011). 4

[49] Y. Liu, K. P. Gummadi, B. Krishnamurthy, A. Mislove, Analyzing facebook privacy settings: user expectations vs. reality, in: Proceedings of the 2011 ACM SIGCOMM conference on Internet measurement conference, IMC '11, ACM, New York, NY, USA, 2011, pp. 61–70.

doi:http://doi.acm.org/10.1145/2068816.2068823.
URL http://doi.acm.org/10.1145/2068816.2068823 4, 18

[50] L. von Ahn, M. Blum, N. J. Hopper, J. Langford, Captcha: Using hard AI problems for security, in: EUROCRYPT, 2003, pp. 294–311. 4, 17

[51] M. Egele, L. Bilge, E. Kirda, C. Kruegel, Captcha smuggling: hijacking web browsing sessions to create captcha farms, in: SAC '10: Proceedings of the 2010 ACM Symposium on Applied Computing, ACM, New York, NY, USA, 2010, pp. 1865–1870. doi:http://doi.acm.org/10.1145/1774088.1774483. 4

[52] C. Hernandez-Castro, A. Ribagorda, Remotely telling humans and computers apart: An unsolved problem, in: J. Camenisch, D. Kesdogan (Eds.), iNetSec 2009 ? Open Research Problems in Network Security, Vol. 309 of IFIP Advances in Information and Communication Technology, Springer Boston, 2009, pp. 9–26.
URL http://dx.doi.org/10.1007/978-3-642-05437-2_2 4

[53] H. Yeend, Breaking CAPTCHA without OCR, http://www.puremango.co.uk/2005/11/breaking_captcha_115/ (November 2005).
URL http://www.puremango.co.uk/2005/11/breaking_captcha_115/ 4

[54] M. Motoyama, K. Levchenko, C. Kanich, D. McCoy, G. M. Voelker, S. Savage, Re: Captchas: understanding captcha-solving services in an economic context, in: Proceedings of the 19th USENIX Conference on Security, USENIX Security'10, USENIX Association, Berkeley, CA, USA, 2010, pp. 28–28.
URL http://portal.acm.org/citation.cfm?id=1929820.1929858 4, 14, 17

[55] F. Devil, Facebook devil: Facebook profile creator, http://www.facebookdevil.com/ (January 2011). 4

[56] M. Broersma, idefense: 1.5 million facebook accounts for sale, http://www.zdnet.co.uk/news/security-threats/2010/04/23/idefense-15-million-facebook-accounts-for-sale-40088751/ (April 2010). 4, 12

[57] M. Motoyama, D. McCoy, K. Levchenko, S. Savage, G. M. Voelker, Dirty jobs: The role of freelance labor in web service abuse, in: Proceedings of the 20th USENIX Security Symposium, 2011.
URL http://www.usenix.org/events/sec11/tech/full_papers/Motoyama.pdf 4, 14

[58] N. FitzGerald, New Facebook worm - don't click da' button baby!, http://fitzgerald.blog.avg.com/2009/11/ (November 2009). 4

[59] A. Mislove, M. Marcon, K. P. Gummadi, P. Druschel, B. Bhattacharjee, Measurement and analysis of online social networks, in: IMC '07: Proceedings of the 7th ACM SIGCOMM Conference on Internet Measurement, ACM, New York, NY, USA, 2007, pp. 29–42. doi:http://doi.acm.org/10.1145/1298306.1298311. 5, 15

[60] Facebook, Facebook graph API, https://developers.facebook.com/docs/reference/api/ (November 2011). 5, 8, 16

[61] M. Huber, M. Mulazzani, E. Weippl, Who on earth is Mr. Cypher? automated friend injection attacks on social networking sites, in: Proceedings of the IFIP International Information Security Conference 2010: Security & Privacy — Silver Linings in the Cloud, 2010. 5

[62] J. Weizenbaum, Eliza — a computer program for the study of natural language communication between man and machine, Commun. ACM 9 (1966) 36–45. doi:http://doi.acm.org/10.1145/365153.365168.
URL http://doi.acm.org/10.1145/365153.365168 6

[63] T. Lauinger, V. Pankakoski, D. Balzarotti, E. Kirda, Honeybot, your man in the middle for automated social engineering, in: Proceedings of the 3rd USENIX conference on Large-scale exploits and emergent threats: botnets, spyware, worms, and more, USENIX Association, 2010, pp. 11–11. 6

[64] S. T. Tong, B. Van Der Heide, L. Langwell, J. B. Walther, Too much of a good thing? the relationship between number of friends and interpersonal impressions on Facebook, Journal of Computer-Mediated Communication 13 (3) (2008) 531–549. doi:10.1111/j.1083-6101.2008.00409.x.
URL http://dx.doi.org/10.1111/j.1083-6101.2008.00409.x 6, 10

[65] J. Leskovec, E. Horvitz, Planetary-scale views on a large instant-messaging network, in: Proceeding of the 17th International Conference on World Wide Web, ACM, New York, NY, USA, 2008, pp. 915–924. doi:http://doi.acm.org/10.1145/1367497.1367620.
URL http://doi.acm.org/10.1145/1367497.1367620 7

[66] E. Kartaltepe, J. Morales, S. Xu, R. Sandhu, Social network-based botnet command-and-control: emerging threats and countermeasures, in: Applied Cryptography and Network Security, Springer, 2010, pp. 511–528. 7

[67] S. Nagaraja, A. Houmansadr, P. Agarwal, V. Kumar, P. Piyawongwisal, N. Borisov, Stegobot: A covert social network botnet, in: Proceedings of the Information Hiding Conference, 2011. 7

[68] C. Lampe, N. Ellison, C. Steinfield, A face(book) in the crowd: social searching vs. social browsing, in: Proceedings of the 2006 20th anniversary conference on Computer supported cooperative work, CSCW '06, ACM, New York, NY, USA, 2006, pp. 167–170. doi:http://doi.acm.org/10.1145/1180875.1180901.
URL http://doi.acm.org/10.1145/1180875.1180901 7

[69] N. B. Ellison, C. Steinfield, C. Lampe, The benefits of Facebook "friends:" social capital and college students' use of online social network sites, Journal of Computer-Mediated Communication 12 (4) (2007) 1143–1168. doi:10.1111/j.1083-6101.2007.00367.x.
URL http://dx.doi.org/10.1111/j.1083-6101.2007.00367.x

[70] C. Lampe, N. B. Ellison, C. Steinfield, Changes in use and perception of facebook, in: Proceedings of the 2008 ACM conference on Computer supported cooperative work, CSCW '08, ACM, New York, NY, USA, 2008, pp. 721–730. doi:http://doi.acm.org/10.1145/1460563.1460675.
URL http://doi.acm.org/10.1145/1460563.1460675 7

[71] N. Bos, K. Karahalios, M. Musgrove-Chávez, E. S. Poole, J. C. Thomas, S. Yardi, Research ethics in the facebook era: privacy, anonymity, and oversight, in: CHI EA '09: Proceedings of the 27th international conference extended abstracts on Human factors in computing systems, ACM, New York, NY, USA, 2009, pp. 2767–2770. doi:http://doi.acm.org/10.1145/1520340.1520402. 8

[72] Facebook, Facebook security, http://www.facebook.com/security (July 2011). 8

[73] BBC News, Socialbots used by researchers to "steal" facebook data, http://www.bbc.co.uk/news/technology-15553192 (November 2011). 8

[74] DSLReports, Trojan horse, and virus FAQ, http://www.broadbandreports.com/faq/trojans/1.0_Trojan_horses (July 2011). 8

[75] C. P. Robert, G. Casella, Monte Carlo Statistical Methods (Springer Texts in Statistics), Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2005. 8

[76] M. Gjoka, M. Kurant, C. T. Butts, A. Markopoulou, Walking in Facebook: A case study of unbiased sampling of osns, in: Proceedings of IEEE INFOCOM '10, San Diego, CA, 2010. 8

[77] D. Lowd, C. Meek, Adversarial learning, in: Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining, KDD '05, ACM, New York, NY, USA, 2005, pp. 641–647. doi:http://doi.acm.org/10.1145/1081870.1081950.
URL http://doi.acm.org/10.1145/1081870.1081950 8, 19

[78] J. B. Grizzard, V. Sharma, C. Nunnery, B. B. Kang, D. Dagon, Peer-to-peer botnets: overview and case study, in: Proceedings of the first conference on First Workshop on Hot Topics in Understanding Botnets, USENIX Association, Berkeley, CA, USA, 2007, pp. 1–1.
URL http://dl.acm.org/citation.cfm?id=1323128.1323129 9

[79] R. Gross, A. Acquisti, H. J. Heinz, III, Information revelation and privacy in online social networks, in: WPES '05: Proceedings of the 2005 ACM workshop on Privacy in the electronic society, ACM, New York, NY, USA, 2005, pp. 71–80. doi:http://doi.acm.org/10.1145/1102199.1102214. 10

[80] R. P. McAfee, Introduction to Economic Analysis, Flat World Knowledge, 2006. 11, 12

[81] Nobody sells gold for the price of silver: Dishonesty, uncertainty and the underground economy (2009). 12, 15

[82] S. Corporation, Symantec report on the underground economy, Online (November 2008). 14

[83] G. Hardin, The tragedy of the commons, Science 162 (1968) 1243–

1248.
URL http://www.sciencemag.org/cgi/reprint/162/3859/1243.pdf 15

[84] G. A. Akerlof, The market for "lemons": Quality uncertainty and the market mechanism, The Quarterly Journal of Economics 84 (3) (1970) 488–500. 15

[85] C. Kanich, N. Weavery, D. McCoy, T. Halvorson, C. Kreibichy, K. Levchenko, V. Paxson, G. M. Voelker, S. Savage, Show me the money: characterizing spam-advertised revenue, in: Proceedings of the 20th USENIX conference on Security, SEC'11, USENIX Association, Berkeley, CA, USA, 2011, pp. 15–15.
URL http://dl.acm.org/citation.cfm?id=2028067.2028082 15

[86] D. Horowitz, S. D. Kamvar, The anatomy of a large-scale social search engine, in: Proceedings of the 19th international conference on World wide web, WWW '10, ACM, New York, NY, USA, 2010, pp. 431–440. doi:http://doi.acm.org/10.1145/1772690.1772735.
URL http://doi.acm.org/10.1145/1772690.1772735 15

[87] F. Walter, S. Battiston, F. Schweitzer, A model of a trust-based recommendation system on a social network, Autonomous Agents and Multi-Agent Systems 16 (2008) 57–74. doi:10.1007/s10458-007-9021-x.
URL http://dx.doi.org/10.1007/s10458-007-9021-x 15

[88] D. N. Tran, F. Chiang, J. Li, Friendstore: cooperative online backup using trusted nodes, in: Proceedings of the 1st Workshop on Social Network Systems, SocialNets '08, ACM, New York, NY, USA, 2008, pp. 37–42. doi:http://doi.acm.org/10.1145/1435497.1435504.
URL http://doi.acm.org/10.1145/1435497.1435504 15

[89] F. Schreiber, S. Martínez, L. Vigil, Sybil, 1981. 16

[90] A. Jøsang, J. Golbeck, Challenges for robust of trust and reputation systems (Sep. 2009). 16

[91] G. Urdaneta, G. Pierre, M. van Steen, A survey of DHT security techniques, ACM Computing Surveys 43 (2), http://www.globule.org/publi/SDST_acmcs2009.html. 16

[92] J. Newsome, E. Shi, D. Song, A. Perrig, The sybil attack in sensor networks: analysis defenses, in: Information Processing in Sensor Networks, 2004. IPSN 2004. Third International Symposium on, 2004, pp. 259 – 268. doi:10.1109/IPSN.2004.1307346. 16

[93] H. Yu, M. Kaminsky, P. B. Gibbons, A. Flaxman, Sybilguard: defending against sybil attacks via social networks, SIGCOMM Comput. Commun. Rev. 36 (2006) 267–278. 16

[94] H. Yu, P. B. Gibbons, M. Kaminsky, F. Xiao, Sybillimit: A near-optimal social network defense against sybil attacks, in: Proceedings of the 2008 IEEE Symposium on Security and Privacy, IEEE Computer Society, Washington, DC, USA, 2008, pp. 3–17. doi:10.1109/SP.2008.13.
URL http://portal.acm.org/citation.cfm?id=1397759.1398053

[95] G. Danezis, P. Mittal, SybilInfer: Detecting sybil nodes using social networks, in: NDSS, 2009.

[96] N. Tran, B. Min, J. Li, L. Subramanian, Sybil-resilient online content voting, in: Proceedings of the 6th USENIX symposium on Networked systems design and implementation, NSDI'09, USENIX Association, Berkeley, CA, USA, 2009, pp. 15–28.

[97] N. Tran, J. Li, L. Subramanian, S. Chow, Optimal sybil-resilient node admission control, in: INFOCOM, 2011 Proceedings IEEE, 2011, pp. 3218 –3226. 16

[98] I. Stoica, R. Morris, D. Liben-Nowell, D. R. Karger, M. F. Kaashoek, F. Dabek, H. Balakrishnan, Chord: a scalable peer-to-peer lookup protocol for internet applications, IEEE/ACM Transactions on Networking (TON) 11 (1) (2003) 17–32. doi:http://dx.doi.org/10.1109/TNET.2002.808407. 16

[99] P. Maymounkov, D. Maziéres, Kademlia: A peer-to-peer information system based on the xor metric, in: Revised Papers from the First International Workshop on Peer-to-Peer Systems, IPTPS'01, Springer-Verlag, London, UK, 2002, pp. 53–65. 16

[100] OpenID Foundation, Promotes, protects and nurtures the OpenID community and technologies, http://openid.net/foundation/ (2009). 16

[101] S. Yardi, N. Feamster, A. Bruckman, Photo-based authentication using social networks, in: Proceedings of the first workshop on Online social networks, ACM, 2008, pp. 55–60. 17

[102] H. Kim, J. Tang, R. Anderson, Social authentication: Harder than it looks, in: Proceedings of the 2012 Cryptography and Data Security conference, 2012. 17

[103] M. Mondal, B. Viswanath, A. Clement, P. Druschel, K. P. Gummadi, A. Mislove, A. Post, Limiting large-scale crawls of social networking sites, in: Proceedings of the ACM SIGCOMM 2011 conference on SIGCOMM, SIGCOMM '11, ACM, New York, NY, USA, 2011, pp. 398–399. doi:http://doi.acm.org/10.1145/2018436.2018487.
URL http://doi.acm.org/10.1145/2018436.2018487 17

[104] C. M. Zhang, V. Paxson, Detecting and analyzing automated activity on twitter, in: Proceedings of the 12th international conference on Passive and active measurement, PAM'11, Springer-Verlag, Berlin, Heidelberg, 2011, pp. 102–111.
URL http://dl.acm.org/citation.cfm?id=1987510.1987521 18

[105] D. Thibeau, Open trust frameworks for open government: Enabling citizen involvement through open identity technologies, http://openid.net/government/ (August 2011). 18

[106] E. Maler, D. Reed, The venn of identity: Options and issues in federated identity management, IEEE Security and Privacy 6 (2008) 16–23. doi:http://doi.ieeecomputersociety.org/10.1109/MSP.2008.50. 18

[107] S.-T. Sun, Y. Boshmaf, K. Hawkey, K. Beznosov, A Billion Keys, but Few Locks: The Crisis of Web Single Sign-On, in: Proceedings of the New Security Paradigms Workshop (NSPW), 2010, pp. 61–72. doi:http://doi.acm.org/10.1145/1900546.1900556.
URL http://doi.acm.org/10.1145/1900546.1900556 18

[108] J. Berson, ZoneAlarm: Creating usable security products for consumers, in: L. F. Cranor, S. Garfinkel (Eds.), Security and Usability: Designing Secure Systems that People Can Use, O'Reilly Media, Inc., 2005, Ch. 27, pp. 563–575. 18

[109] S. L. Feld, The focused organization of social ties, The American Journal of Sociology 86 (5) (1981) 1015–1035.
URL http://www.jstor.org/stable/2778746 18

[110] C. Wilson, B. Boe, A. Sala, K. P. Puttaswamy, B. Y. Zhao, User interactions in social networks and their implications, in: Proceedings of the 4th ACM European conference on Computer systems, EuroSys '09, ACM, New York, NY, USA, 2009, pp. 205–218. doi:http://doi.acm.org/10.1145/1519065.1519089.
URL http://doi.acm.org/10.1145/1519065.1519089 18