Available online at www.sciencedirect.com

ScienceDirect

journal homepage: www.elsevier.com/locate/cose

Computers & Security

Combating the evasion mechanisms of social bots

Yuede Ji, Yukun He, Xinyang Jiang, Jian Cao, Qiang Li [*]

College of Computer Science and Technology, Jilin University, Changchun, China

A R T I C L E   I N F O

Article history:
Received 13 April 2015
Received in revised form 1 December 2015
Accepted 24 January 2016
Available online 4 February 2016

Keywords:
Botnet detection
Social bot
Host behavior
Empirical evaluation
Correlation mechanism

A B S T R A C T

The detection and anti-detection of social botnets constitute an arms race that enables social botnets to evolve quickly. Existing host-side detection approaches cannot easily detect every social botnet. Thus, we propose a new host-side detection approach that can effectively detect existing social bots. The contribution of this study is three-fold. First, we comprehensively analyze the evasion mechanisms used by existing social bots and validate those mechanisms by applying three state-of-the-art detection approaches to our collected traces. To the best of our knowledge, this is the first empirical evaluation of evasion mechanisms used by social bots. Second, based on the insights gained, we propose a new detection approach that incorporates nine newly identified features and two new correlation mechanisms. The new features are classified either as lifecycle or failure based, and the two correlation mechanisms are temporal and spatial correlations. Finally, our experimental results indicate that under various classifiers, our approach can detect existing social bots. Using the random forest classifier, our approach provides about a 0.3% false positive rate, 4.7% false negative rate, 0.963 F-measure value, and 99.2% detection rate. In addition to detecting social bots, our approach yields acceptable detection results for common botnets.

1. Introduction

In recent years, online social networks (OSNs) have gained much popularity (Naessens et al., 2010). Facebook has over one billion active users and each of other 15 OSNs has over 100 million active users. Given this huge number of active users, OSNs have also attracted the attention of cyber criminals who diligently perform malicious activities such as spreading rumor, stealing user privacy, sending spam, spreading phishing email, spreading malware, propagating malicious URL, and launching distributed denial of service (DDoS) attack (Ahn et al., 2011; Avery et al., 2015; Boshmaf et al., 2013; Chandramouli, 2015; Douligeris and Mitrokotsa, 2004; Garfinkel, 2012; Gritzalis et al., 2014; Heikkila, 2009; Lalas et al., 2012; McDonald et al., 2005; Moyano et al., 2013; Safa et al., 2015; Schrittwieser et al., 2012; Spafford, 2010; Suen and Yasinsac, 2005; Zhang et al., 2008). A social botnet refers to a group of social bots under the control of a botmaster who uses OSNs as a command and control (C&C) channel (Baltazar et al., 2009; Zhang et al., 2013). The first social botnet called koobface is discovered on August 3, 2008, which targets most OSN sites such as Facebook, Twitter, and MySpace. Another social botnet called Naz bot is discovered on Twitter in 2009 (Nazario, 2009). In addition, attackers and researchers have designed several social botnet prototypes (Boshmaf et al., 2013; Kartaltepe et al., 2010; Nagaraja et al., 2011; Verkamp et al., 2011). Many traditional botnets are revitalized by propagating on OSN sites. For example, Zeus, which is first detected in 2007, steadily proliferated in 2013 by propagating on Facebook (Anon, 2015a). Another botnet called Pony botnet is found to have stolen two million passwords from Facebook, Twitter, Yahoo, and ADP on December 4, 2013 (Anon, 2015c). In 2014, according to the

[*] *Corresponding author.* Tel.: +86 13331677881.
E-mail address: li_qiang@jlu.edu.cn (Q. Li).

http://dx.doi.org/10.1016/j.cose.2016.01.007
0167-4048/© 2016 Elsevier Ltd. All rights reserved.

security firm Trustwave, Pony botnet steals bitcoins and other digital currencies in the most ambitious cyber attack on virtual money uncovered thus far (Anon, 2015d).

Social botnets have several inherent advantages that enable them to evade common botnet detection approaches (Alotibi et al., 2015; Maybury et al., 2005; Ragsdale et al., 2003; Zhao et al., 2011). First, social botnets abuse trusted and popular websites by acting as C&C servers, which helps to foil traditional botnet server takedown approaches (Nappa et al., 2014). They also exploit popular ports for C&C communication. In their traffic, no suspicious address, domain name, protocol, or port is involved. Therefore, traffic from bots blends in with benign traffic, which helps them effectively evade network traffic detection approaches. Second, most social bots exploit information hiding techniques such as cryptography and steganography to encrypt their commands (Li et al., 2011). Using the hypertext transfer protocol over secure socket layer (HTTPS), social bots can effectively impede content inspection in the network. Third, many social bots mimic user or benign application activities to hide their malicious activities. These three mechanisms allow social bots to effectively evade common host-based botnet detection approaches. The main difference between bots and social bots is that C&C channels of social bots are based on OSN websites. Researchers have proposed many malware analysis methods, some of which are very advanced. Social bots must receive commands from their botmasters via C&C channels. When a social bot is in a malware-analysis environment, it may not be able to fetch commands from its botmaster as usual. Analysis results are highly influenced by the behavior of the botmaster because most social bots that can be used for analysis have been published and their botmasters may have been shut down. As a result, when a social bot runs in a malware-analysis system, it may do nothing except connecting to a social website. Therefore, to detect social bots on end hosts, a novel approach must be developed.

To analyze and detect social botnets, researchers have proposed several constructive approaches, which can be divided into two categories by detection location. The first category detects abnormal host behavior (Chu et al., 2012; Natarajan et al., 2012) such as registry modification, file system information, and system calls. However, social bots perform only a few activities on end hosts. The second category detects abnormal behavior in an OSN (Dietrich et al., 2013; Wagner et al., 2012) based on OSN user information such as messages posted by users and friend requests sent by users. This approach can find malicious OSN accounts or messages. However, social bots have the unique ability to mimic normal users in OSNs, detecting them in OSNs becomes difficult.

The detection of social botnets is an arms race – social botnets continuously evolve to evade new detection features (Stein et al., 2011). Based on our experience, social botnets evolve aggressively and become stronger and more robust. In addition, existing detection approaches from the user side have not kept pace with their evolution. In the causal relationship detection approach, it is difficult to synchronize human activities and network traffic. In addition, it is difficult to quantify the time interval because of many dynamically changing factors such as network delay, operating system delay, and performance of different computers. Moreover, many advanced social bots do not perform malicious activities until they have monitored human activities. Therefore, malicious activities are deliberately mixed with benign human activities. In the approach used by Kartaltepe et al. (2010), the authors implicitly assume that a social bot comprises only one process. However, this assumption does not hold true today because many bots have evolved that exploit multiple processes (Ji et al., 2014; Ma et al., 2012). By dividing their malicious activities across several processes, with each process performing only a portion of the total, the suspicion level can drop to the same as that for benign processes. In addition, social bots are evolving to have more advanced mechanisms such as delayed response to evade detection. Most research has analyzed network flow from social websites to identify suspicious accounts and messages. Therefore, designing effective and efficient host-side social bot detection approaches is an active and urgent research imperative.

Our goal is to provide the first empirical analysis of user-side evasion mechanisms. Subsequently, through an in-depth analysis, we propose a new detection approach for detecting novel social bots. The features used in our research also extend the existing detection approaches of social bots on end hosts. To achieve our goals, we collect the source codes, builders, and execution traces of most existing social botnets. We then analyze existing social bot evasion mechanisms, including four basic and four advanced mechanisms. We validate these mechanisms by applying three state-of-the-art detection approaches to our collected traces. Based on the insights gained, we develop a new detection approach with nine newly identified features and two new correlation mechanisms. The features are classified as being either lifecycle or failure based. The two correlation mechanisms are temporal and spatial correlations, which accumulate historical information and combine different processes, respectively. In the experiment, under different classifiers, our approach can detect existing social bots. Using the random forest (RF) classifier, our approach obtains the following results: about a 0.3% false positive rate (FPR), 4.7% false negative rate (FNR), 0.963 F-measure value and 99.2% detection rate (DR). In addition to detecting social bots, our approach yields acceptable detection results for common botnet.

In summary, the contributions of this study are as follows.

(1) We collect source codes, builders, and execution traces of most existing social botnets and then analyze mechanisms utilized by these social botnets to evade existing detection approaches. We validate the evasion mechanisms by applying three state-of-the-art detection approaches to the collected traces and theoretically analyze reasons for their poor detection results.
(2) Based on the insights gained, we develop a new detection approach consisting of nine newly identified features and two new correlation mechanisms. We classify these new features and other effective features into two categories: lifecycle or failure based. The two new correlation mechanisms, temporal and spatial correlations, are designed to combat advanced evasion mechanisms. Temporal correlation accumulates the historical information, and spatial correlation combines numerous different processes.
(3) We evaluate our approach with respect to the entire set of social bot traces. With the newly identified features

and correlation mechanisms, our approach performs much better than existing approaches. Using the RF classifier, our approach obtains about a 0.3% FPR, 4.7% FNR, 0.963 F-measure value, and 99.2% DR. In addition to detecting social bots, our approach yields an acceptable detection results for common bots.

The rest of this paper is organized as follows. In Section 2, we present related work. In Section 3, we present our empirical evaluation of the evasion mechanisms of evolving social bots. In Section 4, we describe the design of our new detection approach using nine newly identified features and two correlation mechanisms. In Section 5, we present our experiment details and results. Section 6 concludes our study.

2. Related work

In the field of social botnet research, existing work has focused on two main areas: social botnet design and analysis and social botnet detection.

2.1. *Social botnet design and analysis*

The social botnet koobface is first discovered in 2008, as reported by Baltazar et al. (2009). It targets most online social networking (OSN) sites such as Facebook, Twitter, and MySpace. Koobface can perform many malicious activities, such as stealing user information, sending malicious spam via OSN sites and emails, and hijacking Web searches. Another social botnet called Naz bot is discovered on Twitter in 2009 (Nazario, 2009). Erhan et al. analyze the details of Naz bot to reveal its features and C&C mechanisms (Kartaltepe et al., 2010). After the two botnets appear, researchers design and analyze other novel social botnet prototypes (Boshmaf et al., 2013; Singh, 2012; Zhang et al., 2013).

Yazan et al. design a novel social botnet that works on Facebook (Boshmaf et al., 2013). It uses OSN, application programming interface (API) and hypertext transfer protocol (HTTP) requests to communicate with a botmaster. It can perform two types of attacks: those on basic OSN operations (such as reading and writing content on Facebook) and those on social structure operations that can modify the social graph (such as connecting or disconnecting with other accounts). Singh et al. construct a botnet focused on Twitter (Singh, 2012), called *Twitterbot*, which uses individual bots controlled by a botmaster to tweet commands. The twitter botnet can expand virally by following affected Twitter users' friends. The botmaster of *twitterbot* can post tweets with pre-determined keywords. To fetch these tweets from the botmaster account, bots make a query to the Twitter search engine. Then, the bots decode these tweets and conduct corresponding actions. Through real experiments on Twitter and trace-driven simulations, Zhang et al. demonstrated the effectiveness and advantages of exploiting a social botnet for spam distribution and digital influence (Zhang et al., 2013). These authors showed that a social botnet can easily manipulate the digital influence of Twitter users.

Because photographs are frequently used in OSN sites, the use of steganography techniques to construct C&C channels also offered a potential avenue for social botnets, of which Facebot and Stegobot are typical examples (Nagaraja et al., 2011; Verkamp et al.). Facebot works on Facebook using steganographic techniques to hide information in profile pictures (Verkamp et al.). The bot and botmaster join the same Facebook group using an algorithm based on Markov random chains. Then the botmaster scans the profile image of every new member to collect hidden information. Stegobot also uses image steganography to hide information in pictures (Nagaraja et al., 2011). The bots download several recent images of the botmaster and then extract hidden commands. After successfully executing these commands, the bots hide the collected information in images and upload them. The botmaster uses the same method to download bot images and extract hidden information.

Aviad et al. present a method (Elyashar et al., 2013) for infiltrating specific users in targeted organizations using organizational social network topologies and social bots. The authors' main objective is to demonstrate the ease with which the private information of an employee of specific organizations can be infiltrated. First, they crawl on targeted organization website and gather public information regarding employees who have a Facebook user account. Next, they gain sufficient information about Facebook users who have worked in targeted organizations and their connections. Third, they create a Facebook account for every organization they desire to infiltrate. Fourth, they define the platform upon which each social bot will act. Fifth, for each selected targeted user, they mark their friends inside an organization and sent friend requests to each of targeted user's friends. Finally, they send friend requests to 10 targeted users. These results demonstrate how easily adversaries can infiltrate user OSN accounts and obtain full access to valuable information. Faghani and Nguyen propose a new cellular botnet (Faghani and Nguyen, 2012) called SoCellBot, which exploits OSNs to recruit bots and uses OSN messaging systems such as C&C channels. The objective of a SoCellBot botnet is to infect as many smartphones as possible with malware. The medium for spreading the infection is OSN messaging systems. A SoCellBot botnet has three major components: propagation mechanism, C&C channel, and botnet topology maintenance. This study aims to raise awareness of new mobile botnets that exploit OSNs to recruit bots.

Several studies have analyzed social botnets to reveal their inherent features. In recent years, automated programs that act in the place of human agents are increasingly used on Twitter. These Twitterbots or virtual software agents are employed to produce automated posts for various purposes. Many of these programs are malicious. Edwards et al. experimentally test whether a known Twitterbot is perceived differently by users than by a human agent with respect to variables related to perceptions of communication quality (Edwards et al., 2014). The convenience sample comprises 240 undergraduate students enrolled in communication courses at a large U.S. Midwestern research university. The results demonstrate that participants rate the human agent higher than Twitterbot in social and task attraction.

Abokhodair et al. consider one specific social botnet in Twitter (Abokhodair et al., 2015). They present how a social botnet grows over time, how the content of tweets by a social botnet differs from that of regular users in the same dataset, and how the social botnet may have influenced relevant discussions. Their results show that the growth, behavior, and content of this particular botnet did not specifically align with the common conceptions of botnets.

Freitas et al. try to understand the infiltration strategies of social bots on Twitter (Freitas et al., 2015). They create 120 social bot accounts having different characteristics and using different strategies. Moreover, they use a 2*k* factorial design to quantify the infiltration effectiveness of the different bot strategies studied. The results show that even social bots that employ simple automated mechanisms can successfully infiltrate networks.

Bots can be benign or malicious and they may aim to persuade, smear, or deceive. Ferrara et al. discuss the characteristics of modern sophisticated social bots, and how their presence can endanger OSNs (Ferrara et al., 2015). One of the greatest challenges in bot detection in social media is to understand what modern social bots can do. Early bots mainly automatically perform one type of activity, i.e., posting content. Today, Twitter bots have become increasingly sophisticated, making their detection more difficult. The boundary separating human-like and bot-like behaviors is now fuzzier.

2.2. *Social botnet detection*

Based on their research, Hwang et al. (2012) publish several articles to highlight the growing seriousness of issues caused by social botnets. Boshmaf et al. observe that defending against malicious bots raises a set of unique challenges related to Web automation, online offline identity binding, and usable security (Boshmaf et al., 2012). They concluded that a successful infiltration campaign has three serious security implications. First, the social structure of the target OSN can be compromised and polluted with a large number of non-genuine social relationships. Second, the adversary can breach the privacy of infiltrated users by harvesting large amounts of private data. Finally, the adversary can exploit infiltrated OSNs to spread misinformation.

2.2.1. *Server-side detection*

Several concerted efforts have been made to defend against malicious OSN accounts (Boshmaf et al., 2015; Cao et al., 2014; Yang et al., 2014). Yang et al. propose a Sybil account and character detection approach using a dataset from Renren (Yang et al., 2014). They study Sybils' link creation behaviors, fine-grained behaviors, and behind-the-scenes collusion activities between large groups. Cao et al. observe that, typically, malicious accounts have loosely synchronized actions in an OSN (Cao et al., 2014). The authors cluster user accounts based on the similarity of their actions and detect similarly behaving large groups of malicious accounts. Boshmaf et al. design a scalable OSN fake account detection approach, Integro, which utilizes a user ranking scheme (Boshmaf et al., 2015).

Wagner et al. study data from the social bot Challenge 2011, in which three teams implemented a number of social bots to influence user behavior on Twitter (Wagner et al., 2012). They develop models to identify susceptible users in a set of targets and to predict users' levels of susceptibility, according to three different feature groups (network, behavioral, and linguistic). The results suggest that susceptible users tend to use Twitter for conversational purposes and tend to be more open and social, by virtue of their communications with many different users, their use of more social words, and their demonstration of more affection than non-susceptible users.

Chu et al. propose an approach for classifying human, bot, and cyborg accounts on Twitter (Chu et al., 2012). The authors analyze a collection of over 500,000 accounts to observe differences between human, bots, and cyborgs with respect to tweeting behavior, tweet content, and account properties. Based on their measurement results, they propose a classification system with four components – an entropy-based component, a spam-detection component, an account-properties component, and a decision maker. They are then able to differentiate between humans, bots, and cyborgs based on various combinations of these four components.

2.2.2. *Host-side detection*

Tan et al. propose an approach for detecting spam in user-generated content on social networks (Tan et al., 2012). They find that spammers exhibit unique non-textual patterns, such as posting activities, advertised spam link metrics, and spam hosting behaviors. Based on these non-textual patterns, they propose an offline detection approach utilizing several classification methods. Subsequently, they propose the runtime spam posts detection approach.

Chu et al. employ behavioral biometrics, including mouse and keystroke dynamics, to distinguish between human and blog bots (Chu et al., 2013; Karlesky et al., 2014). They develop a passive, Webpage-embedded logger to collect user input activities on a real, active blog site. By measuring and characterizing the biometric features of user input data, they discover a number of critical differences between humans and blog bots in how each surfs Webpages and posts comments. Subsequently, they design a detection system consisting of two components – a Webpage-embedded logger and a server-side detector. The logger continuously monitors user activities and sends them to the server-side detector. The detector uses a machine-learning-based classifier, which is tuned with training data for binary classification. The main disadvantage of this approach is that it relies on software loaded onto the client browser, which can be difficult to implement and cannot be generalized for all users due to confidentiality constraints.

Francisco et al. suggest that human actions have an inherent pseudo-periodicity mixed with random (and sometimes chaotic) actions, which are almost impossible to emulate/simulate (Brito et al., 2013). However, at the same time, it is easy to differentiate this unique human behavior uniqueness from other behavioral patterns. Therefore, they propose a methodology that jointly analyzes multiple scales of user interactions within a social network, and discriminates between the characteristic behaviors of humans and bots within a social network.

Pieter et al. propose an approach for detecting social botnet communication by monitoring user activity (Burghouwt et al., 2011). These authors suggest that any communication with social media is suspicious if it is not generated by human activity. By measuring the causal relationship between network traffic and human activities, suspicious activity and a potential social bots can be detected. However, the causal relationship between human activities and network traffic is difficult to synchronize. First, it is difficult to quantify the time interval as there are many dynamically changing factors, such as network delay, operating system delay, and the performance of different computers. Second, many advanced social bots do not perform malicious activities unless they are monitoring human activities. Therefore, malicious activities are mixed in with benign human activities.

Natarajan et al. develop a detection scheme to detect StegoBots (Nagaraja et al., 2011), and analyze the different entropies of images to show that images are generally sensitive to embedding. Based on their analysis, they select efficient features to construct the feature set. The authors further propose an ensemble of classifiers for classifying vulnerable images from social networks.

Erhan et al. analyze Naz bot to determine its social bot features and to improve the detection of social bot C&C mechanisms (Kartaltepe et al., 2010). The authors propose both server- and host-side detection mechanisms. In server-side detection, they suggest that if botmasters intend to use social networks for their C&C channels, they may encode their commands textually. At this point, to determine whether a message is suspicious, the authors distinguish between encoded and plain texts and follow un-encoded links to their destination. However, their assumption is not sufficiently robust to cover most situations. First, social bots may not use text to encrypt commands, but instead might use steganography or other encryption methods. Second, there are numerous textual encryption methods, and social bots can even develop their own encryption methods, thus making it very difficult for server-side detection mechanisms to cover all possible encryption methods. In host-side detection, the authors use three social bots attributes self-concealment, dubious network traffic, and unreliable provenance to detect suspicious social bot processes. They make the implicit assumption that a social bot only do single process in the host. However, this assumption may not hold because, currently, many bots are evolving for use in multiple processes.

3. Understanding evasion mechanisms

In this section, we provide details regarding existing evasion mechanisms. First, we introduce data collection strategies and results. Next, we describe the evasion mechanisms utilized by social botnets. Finally, we validate these mechanisms by applying three state-of-the-art detection approaches to the traces collected for this study.

3.1. *Data collection*

Our data collection strategies comprise two parts – social botnet collection and trace collection. It can be challenging to collect social botnet source code or builders, since hackers use complicated encryption mechanisms to pack the source code and spread bot binaries to infect hosts. In addition, the researchers who have source codes or builders are not allowed to share them with the public given the constraints associated with academic ethics. After several attempts, we obtain source codes for most existing social botnets, including Twitterbot (Singh, 2012), Twebot (Burghouwt et al., 2013), Yazanbot (Boshmaf et al., 2013), Nazbot (Kartaltepe et al., 2010), wbbot (Ji et al., 2014), and fbbot. The author of Twitterbot shares its source code with us (Singh, 2012). The author of a social botnet detection approach shares the Twebot builders with us (Burghouwt et al., 2013). Based on the description provided by Boshmaf et al. (2013), we reproduce it and name it Yazanbot. Based on the analysis of Nazbot in Kartaltepe et al. (2010), we reproduce it and named it fixnazbot. We also design wbbot, which acts on Sina Weibo (Ji et al., 2014). In addition to wbbot, we use similar techniques to design fbbot based on Facebook.

Table 1 – Collected traces.

Trace	Duration	Size
Twitterbot	24 h	8.36 GB
TWebot	18 h	2.77 GB
Yazanbot	24 h	7.36 GB
FixNazbot	24 h	4.99 GB
Wbbot	18 h	11.5 GB
Fbbot	5 h	4.65 GB
Total	113 h	39.63 GB

In our trace collection, we set up VMWare virtual machines running Windows XP. We use Process Monitor to record Registry and File operations, Microsoft Network Monitor to collect network traffic, and a python hook to record mouse and keyboard events. When the bots are running, we request users to operate the virtual machine as they would normally. Both benign and malicious behaviors ware captured, and Table 1 shows the details of these traces. For the purposes of academic research, we would like to share the social botnet source codes and traces.[1]

3.2. *Evasion mechanisms of social bots*

Social botnets utilize many distinctive and deceptive evasion mechanisms, which represent the key challenges associated with social botnet detection. We classify these mechanisms into two categories – basic and advanced mechanisms.

3.2.1. *Basic evasion mechanisms*

Although social botnets differ, they share some common detection evasion mechanisms to evade, such as propagating on OSN sites, and performing fewer pre-defined host behaviors than traditional botnets. We define these as basic evasion mechanisms, as shown in Table 2, and utilize the notation E_b^i to denote the ith basic evasion mechanism.

3.2.1.1. $\mathbf{E_b^1}$: *propagating on OSN sites*. OSN sites present several natural advantages to the propagation of malicious bot binaries, such as the trust between friends, fast propagation speed, and encryption. First, in OSN sites, users can share their ideas, pictures, or videos with friends. Since this content comes from our friends, we implicitly trust them. Therefore, social bots such as koobface can exploit this trust to infect more users (Baltazar et al., 2009). Second, messages in OSN sites propa-

[1] http://pan.baidu.com/s/1c0fix00.

Table 2 – Social botnet evasion mechanisms.

Notation	Description	Related social bots
E_b^1	Propagating on OSN sites	All
E_b^2	Less pre-defined host behaviors	All
E_b^3	A lot of OSN-related host behaviors	All
E_b^4	Exploiting popular OSN sites as C&C server	All
E_a^1	Exploiting multiple processes	Fbbot, Wbbot
E_a^2	Mimicking OSN activities of users or automation applications	FixNazbot, Twebot, Twitterbot, Fbbot, Wbbot
E_a^3	Multiple command encryption mechanisms	FixNazbot, Wbbot Twebot, Fbbot
E_a^4	Delayed response	Fbbot, Wbbot

gate rapidly through retweets (Yu et al., 2013), so malicious URLs also propagate fast and cover a wide range (Cao et al., 2016). Third, the URLs of OSN sites are transferred into shorten URLs, such as those of Twitter and sina weibo. This enables attackers to hide the true URL domain, and thereby prevent the OSN sites from effectively applying blacklists to filter out these malicious URLs (Wang et al., 2013). In addition, social bots use deceptive social engineering techniques to induce certain behaviors by users, such as the provocative messages employed by koobface.

3.2.1.2. $\mathbf{E_b^2}$: *fewer pre-defined host behaviors*. Social bots perform fewer pre-defined host behaviors than traditional bots. Because the host behaviors of traditional bots have been thoroughly analyzed in previous studies (Kolbitsch et al., 2009; Park and Reeves, 2009; Shin et al., 2012), social bots performing similar behaviors can be easily detected. In this situation, however, social bots possess the most essential behaviors, such as the capability to modify a bootstrap list to cause the social bots to start along with the system or browser.

3.2.1.3. $\mathbf{E_b^3}$: *many OSN-related host behaviors*. Although social bots have fewer pre-defined host behaviors, they have many OSN-related host behaviors, such as checking Internet cookies to track a user's OSN activities. Social bots attempt to connect with the botmaster by various mechanisms. For example, Nazbot visits the Really Simple Syndication (RSS) of some specific user accounts to establish a C&C channel, and Stegobot shares images to establish a C&C channel using image steganography to hide sensitive information (Nagaraja et al., 2011). Many benign software applications, such as OSN-related applications, have similar and potentially even more behavioral patterns than social bots, which makes detection more difficult.

3.2.1.4. $\mathbf{E_b^4}$: *exploiting popular OSN sites as C&C servers*. Traditional botnets that use Internet Relay Chat (IRC) or HTTP as THEIR C&C channel face the challenge of single-point failure, in which the server is detected and shut down. However, social botnets can easily overcome this challenge since their servers are popular OSN sites, which are certainly on the white list. This evasion mechanism is fatal to traditional botnet detection approaches since there are no anomalous addresses, domain names, protocols, or ports, and a large fraction of the legitimate traffic of normal computers includes visits to OSN sites.

3.2.2. *Advanced evasion mechanisms*

In the arms race between social bots and the development of detection approaches, social bots have evolved to include new evasion mechanisms and existing host-side detection approaches have not been able to keep pace with them. We consider the new evasion mechanisms to be advanced evasion mechanisms, as summarized in Table 2 and we use the notation E_a^i to denote the ith mechanism.

3.2.2.1. $\mathbf{E_a^1}$: *exploiting multiple processes*. Existing social bots have evolved to exploit multiple processes to evade detection, such as fbbot and wbbot, as shown in Table 2. Social bots assign malicious behaviors to several processes, and each process performs several malicious behaviors. Consequently, the suspicion level of each process can drop to the same level as that for a benign process. Existing detection approaches mainly focus on one process, which can be easily evaded by multiple process bots (Burghouwt et al., 2011; Chu et al., 2012; Kartaltepe et al., 2010). Multiple process bots have been analyzed in several studies. Ma et al. present a multiple process mechanism to evade existing behavior-based malware detection by dividing a malware into multiple "shadow processes" (Ma et al., 2012). Ji et al. present a multiple process mechanism for evading behavior-based bot detection approaches at the end host (Ji et al., 2014).

3.2.2.2. $\mathbf{E_a^2}$: *mimicking OSN activities of users or automation applications*. Existing social bots not only perform malicious behaviors, but also try to perform behaviors to decrease the suspicion level. Taking Twitterbot as an example (Singh, 2012), it can not only fetch botmaster's tweets, but also update its own status like any normal user, using the status update component. Social bots can automatically update their status on OSN sites – such as Facebook and Twitter – using random message applications, such as *I Heart Quotes* (Anon, 2015a). With these human-mimicking behaviors, social bots can effectively confuse existing detection approaches. Social bots also attempt to mimic the behaviors of OSN-related applications, such as Twitterdeck and weibo desktop. Based on our observations, these applications perform OSN activities more frequently than those associated with social bots, making it difficult to distinguish social behaviors bots from the large range of OSN application behaviors. This evasion mechanism essentially confuses existing detection approaches.

3.2.2.3. $\mathbf{E_a^3}$: *multiple command encryption mechanisms*. To better hide information, social bots encrypt messages using various encryption mechanisms. They hide commands, and execution results in encrypted messages, which can be either a normal sentence or messy code. For example, Nazbot uses Base64-encoded messages to hide its commands, wbbot and fbbot use the Data Encryption Standard (DES), and Koobface uses simple bitwise-ADD and bitwise-OR operations (Baltazar et al., 2009). Although the encryption algorithms are simple, the decryption process can be time consuming. It is difficult to predict the encryption algorithms used by social bots, and with so many encryption algorithms, the huge costs of decryption will pose an impossible challenge. Therefore, detection approaches based on the command signature of the text message are inefficient.

3.2.2.4. $\mathbf{E}_{\mathbf{a}}^{\mathbf{4}}$: *delayed response*. Most existing social bot detection approaches analyze the behaviors within a small time interval or time window, and attackers can set a random time delay between different behaviors. For example, social bots can wait for a random amount of time before executing a received command (Al-Hammadi and Aickelin, 2010; Zeng, 2012). Therefore, receiving commands and the execution of tasks can be split into different time windows, which can confuse detection approaches. However, if you set a sufficiently wide time window to solve this problem, it will result in a heavy overload because of large amounts of stored information. Given that most host behaviors are similar to human operations or OSN-related applications, social bots can utilize this mechanism well.

3.3. *Validation of evasion mechanisms*

Since social bots attempt to use the advanced evasion mechanisms to evade existing detection approaches, we focus on validating them in this section. First, we reproduce three existing detection approaches (Burghouwt et al., 2011; Chu et al., 2013; Kartaltepe et al., 2010), and evaluate them based on the traces collected (to better explain our result, we label the work of Kartaltepe et al. (2010) as *A*, Chu et al. (2013) as *B*, and Burghouwt et al. (2011) as *C*). Subsequently, we theoretically analyze the advanced evasion mechanisms with respect to these detection approaches.

3.3.1. *Implementation of existing detection approaches*

We replicate three existing detection approaches reported in the works of Burghouwt et al. (2011), Chu et al. (2013), and Kartaltepe et al. (2010), which are representative host-side social botnet detection approaches. Using their assumptions and datasets, we obtained fairly good detection results for social botnets. So, using these detection approaches to validate existing social botnets can clearly identify evolving trends. We also compared our approach with theirs to test the effectiveness of our approach in detecting social botnets.

In Kartaltepe et al. (2010), the authors propose two countermeasures for social bots, one a server-side, and the other a client-side. Since our approach involves host-side detection, we reproduced their client-side detection approach. In Chu et al. (2013), the authors propose a blog-bots detection approach using behavioral biometrics. In Burghouwt et al. (2011), the authors propose a social botnet communication detection approach by monitoring user activity. This detection approach is improved in Burghouwt et al. (2013) and named CITRIC. The authors of CITRIC graciously shared its source code with us.

Detection approach A uses three attributes – self-concealing, dubious network traffic, and unreliable provenance – to determine whether a process is benign or that of a social bot (Kartaltepe et al., 2010). A process *P* has the self-concealing attribute (P_{sc}) if it does not have a graphical user interface (P_{gui}) and human computer interaction (P_{hci}). It has the dubious network traffic attribute (P_{dnt}) if it has social network traffic requests (P_{snr}) and encoded text processing (P_{etp}) or suspicious file downloading (P_{sfd}). It has the unreliable provenance attribute (P_{up}) if it has self-reference replication (P_{srr}) or dynamic code injection (P_{dci}), and also lacks a verified digital signature (P_{vds}). Based on these three attributes, a process is a social network-based bot (P_{snbb}) if it is either self-concealing or has an unreliable provenance (or both), and has dubious network traffic. The formalizations of these three attributes and final a detection rule are presented in Equation (1).

$$\begin{aligned}
P_{sc} &= (\neg P_{gui}) \wedge (\neg P_{hci}) \\
P_{dnt} &= P_{snr} \wedge (P_{etp} \vee P_{sfd}) \\
P_{up} &= (P_{srr} \vee P_{dci}) \wedge (\neg P_{vds}) \\
P_{snbb} &= (P_{sc} \vee P_{up}) \wedge P_{dnt}
\end{aligned} \tag{1}$$

Detection approach B uses behavioral biometrics to identify blog bots (Chu et al., 2013). They select five user input actions to represent user behaviors – keystroke, point, click, point-and-click, and drag-and-drop. Based on their observations, for each user input action, eight effective classification features are extracted – duration, distance, displacement, displacement angle, average speed, move efficiency, virtual key value, and timing entropy. They select C4.5 algorithm, and use ten-fold cross validation on the collected data to evaluate their detection approach on collected data.

Detection approach C detects covert botnet C&C channels by the causal analysis of traffic flows (Burghouwt et al., 2013). They define the direct cause of a traffic flow as the event that ultimately triggers the flow. They identify four benign events – traffic events, user events, process events, and server events. Traffic events are the most common direct causes of new traffic events. For instance, the newly received IP-address in a DNS reply is used as the destination of a new HTTP flow. User events are user input actions via a mouse, keyboard, touching screen, or other input devices. Process events are state transitions of software processes that generate automatic flows, such as software updates. Server events are flows originating from other hosts to the observed host. A traffic flow is classified as anomalous if the direct cause is a process event, and the remote address or hostname is not white-listed.

3.3.2. *Validation result analysis*

We use false positive (FP) rate and false negative (FN) rate to denote the detection results. An FP represents a benign process that is misclassified as a social bot, while an FN represents a social bot that is misclassified as a benign process. We select a duration of one hour for the collected traces to evaluate the detection approaches following their experiments, and the detection results are presented in Table 3. Three detection approaches all have high FN rates that are above 40%, while all of them have low FP rates (FP rate for A is 10.8%, B is 10.2%, and C is 11.5%). These results indicate that the three approaches can handle benign processes fairly well, while it can be challenging to handle the evolving social bots. Next, we analyze the validation results of each detection approach in detail.

Table 3 – Detection results.

Traces	A		B		C	
	FPr	FNr	FPr	FNr	FPr	FNr
Fbbot	12%	64.3%	10.3%	42.9%	9.2%	71.4%
FixNazbot	9.1%	0	10%	62.5%	10.8%	25%
Twebot	9.4%	50%	10.2%	62.5%	13.4%	12.5%
Twitterbot	9.7%	0	10.4%	100%	9.7%	33.3%
Wbbot	10.3%	54.5%	10.2%	36.4%	14.6%	81.8%
Yazanbot	11.4%	50%	9.8%	66.7%	10.9%	50%
Total	10.8%	44%	10.2%	54%	11.5%	52%

The 10.8% FP rate of detection approach A indicates that it can identify benign processes fairly well. However, its 44% FN rate indicates that it has a very low detection rate for social bots. Regarding the specific social bots, approach A has a 0% FN rate for Nazbot and Twitterbot, a 50% FN rate for Yazanbot, a 54.5% FN rate for wbbot, and greater over 60% toward fbbot. The reasons for these high FN rates can be stated as follows: (1) Although the detection attributes seem accurate, they are difficult to quantify. Using encoded text processing (P_{etp}) as an example, there are many text encoding or encryption methods. If the method used by social bots is known, then it is straightforward matter to capture it. For example, we can capture Nazbot since we know that it uses Base64-encoded method. However, it is impossible to decrypt, if the encryption method is not known. As shown in Table 2, fbbot, Nazbot, Twebot, and wbbot use various encryption mechanisms (E_a^3), which result in correspondingly high FN rates. (2) The detection attributes are deterministic, so if the social bots evade one or several rules, they are able to evade the whole detection approach. For example, they may use the advanced evasion mechanism E_a^4 (delayed response) to split their behaviors into different time windows. Once they have evaded the detection rule social network request (P_{snr}), they can evade the whole detection approach.

Detection approach B also demonstrates a fairly good FP rate of 10% and a high FN rate of 54%. It uses behavioral biometrics to classify blog bots, but social bots have several mechanisms by which they operate on OSN sites, such as OSN APIs, RSSs, and the Web Automation Test (WAT). This approach can capture the behaviors of the WAT mechanism, but the other mechanisms cannot be captured. Since Nazbot and Twebot use the RSS mechanism to receive encrypted commands, Yazanbot uses Facebook open API to operate on Facebook, and Twitterbot uses Twitter open API to operate on Twitter, approach B is unable to capture their behavioral biometrics, which results in its high FN rate. However, fbbot and wbbot use WAT to operate on OSN sites, which can be captured by approach B. Although they are captured, fbbot and wbbot both use the advanced evasion mechanism: E_a^2, to mimic the OSN activities of users or applications. Therefore, approach B can achieve only a 42.9% FN rate for fbbot, and 36.4% for wbbot. We note that approach B mainly focuses on blog bots, and especially on the human mimic bot and replay bot. Although blog bots intersect with social bots to some degree, they exhibit different behaviors. We confirm that approach B can yield fairly good detection results for blog bots, while this detection approach must be significantly improved to detect similar social bots.

Detection approach C also has a fairly good FP rate of 11.5%, and also a high FN rate of 58%. This approach C faces several challenges in detecting benign processes: (1) There may be some delays between a user input and network traffic, such as an operating system delay, computer performance, or network delay. If the delay splits the user input and network traffic into different time intervals, the traffic may be identified as malicious. (2) Automatic applications – instant message applications, e-mail check applications, and automatic update applications – cause a lot of confusion in approach C. In particular, the OSN-related automatic applications, such as Tweet deck and Facebook blaster, can be easily misclassified as social bots.

Approach C faces several challenges with respect to evolving evasion mechanisms. The first challenge is the advanced evasion mechanism, that exploits multiple processes (E_a^1). Approach C makes an implicit assumption that a social bot is a single process. However, if social bots divide theire behaviors into multiple processes, approach C can miss some malicious processes. Since fbbot and wbbot exploit this evasion mechanism, the FN rates for approach C are 71.4% and 81.8%, respectively. From these results, we see that this evasion mechanism confuses to approach C. The second challenge is another advanced evasion mechanism, which while possible (Ji et al., 2014), has not yet been deployed by existing social bots. This mechanism is the one in which social bots will not perform malicious or OSN activities until they have monitored human activities or even human OSN activities. In this manner, malicious behaviors are mixed with benign human activities, and thus the traffic of social bots is mixed with the large volumes of benign traffic. This evasion mechanism confounds another assumption of approach C, which is that bot-originated traffic is not synchronized with user activity. The FN rates for detecting other social bots are also somewhat high, e.g., 50% for Yazanbot and 33.3% for Twitterbot.

From the above validations, we see that existing social bots evolve to perform complicated and advanced evasion mechanisms, which can confuse existing host-side social bot detection approaches. Therefore, designing new host-side defense mechanisms against social bots is an urgent tasks in the field of botnet detection.

4. New detection approach

In this section, we describe the development of our new detection approach, which involves 18 features, nine of which are previously existing features and the other nine are newly identified. There are also two correlation mechanisms in our approach – spatial correlation and temporal correlation.

4.1. *Newly identified features*

A robust feature should either be difficult or expensive for a malicious entity to evade. A feature is difficult to evade if it requires an intrinsic change to perform malicious activities; whereas, a feature is expensive to evade if it requires significant money, time, or other resources (Yang et al., 2013). With respect to the special characteristics of evolving social bots, we identified nine new features and classified them into two categories – lifecycle- and failure-based.

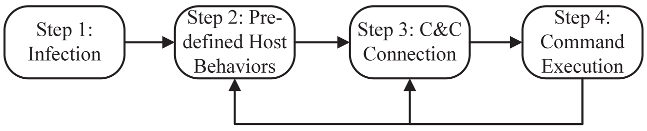

Fig. 1 – Lifecycle of social bots.

4.1.1. *Lifecycle-based features*

To extract features that are ubiquitous in all social bots, we analyze them with respect to their lifecycle. The lifecycle of social bots can be divided into four steps as shown in Fig. 1. In the first step, infection, social bots use the successful infection mechanisms of conventional botnets, including malicious URLs in an email, unwanted malware downloading, and cracked softwares installation (Silva et al., 2013). Social botnets can also propagate on OSN sites using malicious URLs (Baltazar et al., 2009). After infection, social bots perform some pre-defined host behaviors, such as modifying the bootstrap list of a system, or checking Internet cookies. Consequently, social bots work to build a C&C connection with the botmaster. Finally, the bots execute commands received from the botmaster. If some of the commands are host-related, social bots return to the second step. If, instead, some of the commands are OSN or network related, social bots return to the third step.

While it is important to analyze the infection mechanisms to prevent a host from being infected, our work mainly focuses on detecting the behavior of social bots, so we omit the infection step here and make an assumption that social bots have already infected the host using existing infection mechanisms. In the following three steps, we identify several new social bot features.

4.1.1.1. Pre-defined host features. After successful infection, social bots attempt to perform malicious behaviors that will improve their robustness. These behaviors are different from those of conventional bots since they mainly target OSN sites. In this step, we use the four features shown in Table 4 to identify the social bots. The first three features – modifying the bootstrap list of a system, stealing sensitive information, and process injection – are analyzed in conventional botnet detection (Silva et al., 2013), and F_3 is used in detection approach A. We identify a new feature, checking Internet cookies, which is a significant social bots behavior. Based on our observations, most social bots check Internet cookies to identify the OSN sites used by the user. For example, the infection binary of koobface – koobface downloader – checks the Internet cookies first. Then, it reports these OSN site cookies to the koobface C&C server (Baltazar et al., 2009). Depending on the social network cookies found, koobface C&C server determines the additional components that are required for download by the koobface downloader.

4.1.1.2. C&C connection features. Social bots have many different mechanisms for establishing C&C connections. Koobface connects to the C&C server through the HTTP to establish the C&C channel (Baltazar et al., 2009), which is the mechanism also utilized by Yazanbot (Boshmaf et al., 2013). Nazbot visits the RSS of some specific user accounts to establish the C&C channel (Kartaltepe et al., 2010). Stegobot shares images to establish the C&C channel using image steganography to hide sensitive information (Nagaraja et al., 2011). Facebot also utilizes image steganography to establish the C&C channel by hiding sensitive information in user profile pictures (Verkamp et al.,). wbbot and fbbot visit some specific OSN user profiles to build the C&C connections.

In light of these facts, we use the seven features shown in Table 5 to identify the social bots C&C connections. The first two features – F_5 the number of unique IPs contacted, and F_6 the number of unique domains queried – are used currently in conventional botnet detection (Ji et al., 2015; Zeng, 2012). Feature F_7, the number of visited OSN IPs, is used in detection approaches A, C. In addition to these, we also identify three new detection features. If the botmaster utilizes OSN accounts to control social bots, a newly infected host will attempt to visit the botmaster account. Thus we can identify the new feature F_8, the number of visited OSN accounts, and record this behavior. In several existing social bots, the botmasters publish their commands as encrypted messages on OSN sites. In this design, social bots will try to grab the latest messages of the botmaster account. To combat this behavior, we identify the new feature F_9, the number of visited OSN user messages. As noted above, several social bots utilize image steganography to hide commands and other information. Therefore, we also identify the new feature F_{10}, the number of visited OSN user messages, as part of our strategy to combat this evasion mechanism.

4.1.1.3. Command execution features. Social bots employ different command receiving methods than conventional bots. In conventional bots, the command receiving method is based on a "push" model, whereby bots passively wait for the botmaster to send commands. In a social botnet, however, the command receiving method is based on a "pull" model, whereby bots proactively try to grab commands from the botmaster. After the social bots receive these commands, they attempt to execute them. As shown in Fig. 1, some of the commands are host related, and some are OSN or network related. After successful command execution, social bots send the gathered

Table 4 – Pre-defined host behaviors.

Index	Feature	Used or new
F_1	Modifying bootstrap list of system	Used
F_2	Stealing sensitive information	Used
F_3	Process injection	Used
F_4	Checking Internet cookies	New

Table 5 – C&C connection features.

Index	Feature	Used or new
F_5	Number of unique IPs contacted	Used
F_6	Number of unique domains queried	Used
F_7	Number of visited OSN IPs	Used
F_8	Number of visited OSN accounts	New
F_9	Number of visited OSN user messages	New
F_{10}	Number of visited OSN user pictures	New

Table 6 – Command execution features.

Index	Feature	Used or new
F_{11}	Number of opened ports	Used
F_{12}	Number of uploaded OSN messages	New
F_{13}	Number of uploaded OSN pictures	New

information or execution results to the botmaster. The mechanism by which bots communicate with the botmaster is similar to that for establishing a C&C connection, except that the connection is reversed. Social bots attempt to publish the information as a new message or comment with regard to the corresponding message from the botmaster. We identify a new feature F_{12}, the number of uploaded OSN messages, to identify this evasion mechanism. Some social bots also publish images using image steganography. so, we can identify another new feature F_{13}, the number of uploaded OSN pictures. Note that some social bots still use HTTP-based C&C channels, thus we use the feature F_{11} the number of opened ports, which is effective in combating conventional bots (Zeng, 2012). These three command execution features are shown in Table 6.

4.1.2. *Failure-based features*

While existing botnets do not depend exclusively on a single C&C server, they strengthen their robustness using domain flux, frequent updates of C&C servers, and a high level of redundancy (Neugschwandtner et al., 2011). For example, the koobface uses about one hundred C&C servers running on compromised hosts. However, botnets can be targeted by successful node enumeration, infiltration, or take-down operations. Some C&C server URLs embedded in bot binaries are invalid, which can cause several network failures. To combat social botnets specifically, most OSN sites have their own malicious accounts or spam detection systems, such as the Facebook Immune System (Stein et al., 2011). The detection system can identify some accounts controlled by the botmaster, so, it is clear that social bots can generate some OSN-related failures. Update of OSN sites can also cause social bots to generate failure information. For example, sina weibo upgrades to the V6 version on October 13, 2014, caused the original version of wbbot to generate a lot of failure information. The same situation happens in social bots when OSN sites conduct open API upgrades. However, benign OSN applications or human operations usually do not generate network or OSN failures. Based on these heuristics, we can identify failure-based features to combat social bots.

Here, we identify three new failure-based features as shown in Table 7, F_{16} the number of failed OSN queries, F_{17} the number of failed OSN domains and F_{18} the number of failed visited OSN accounts. Note that some social bots still use HTTP-based C&C channels, so we add two existing features F_{14}, the number of failed IPs, and F_{15}, the number of failed domains.

Table 7 – Failure-based features.

Index	Feature	Used or new
F_{14}	Number of failed IPs	Used
F_{15}	Number of failed domains	Used
F_{16}	Number of failed OSN queries	New
F_{17}	Number of failed OSN domains	New
F_{18}	Number of failed visited OSN accounts	New

4.1.3. *Feature robustness*

The robustness feature is based on the activities of a social botnet and is assigned a robustness value. This feature should either be difficult or expensive for a malicious entity to evade. A feature is difficult to evade if it requires an intrinsic change to perform malicious activities; whereas a feature is expensive to evade if it consumes significant amounts of money, time, or other resources.

We classified robustness as being either low, medium, or high. Feature F4, the checking of Internet cookies, is carried out by social bots to track the OSN sites that users have visited. Attackers must access Internet cookies if they want to capture users habits. However, not all social bots must access information about user habits. Therefore, we consider the F4 robustness value to be low. Feature F8 is the number of visited OSN accounts, F9 is the number of visited OSN user messages, and F10 is the number of visited OSN user pictures. When a social bot engages in activities on an OSN, it must login to the OSN and visit the OSN account. However, it may communicate through OSN user messages or encrypt messages in OSN user pictures. Therefore, we considered the F8 robustness value to be high and those of F9 and F10 to be medium. For the same reason, we consider the robustness value of F12 and F13 to be medium. F16 is the number of failed OSN queries, F17 is the number of failed OSN domains, and F18 is the failed visits to OSN accounts. We believe that not all social bots need to query information on the OSN, although most do have to visit the OSN domain and OSN accounts, and this procedure may produce a lot of information on failed domains and visits to accounts. Thus, we consider the robustness value of F16 to be low and those of F17 and F18 to be high.

4.2. *Newly identified correlation mechanisms*

Although we identify several new efficient features, social bots can still use the above mentioned advanced evasion mechanisms, especially E_a^1, exploiting multiple processes and E_a^4, delayed response. To combat these, we also designed two new correlation mechanisms – spatial correlation and temporal correlation.

In contrast to the detection features, these mechanisms aim to update the current process feature vector by correlating related feature vectors. First, we extract the feature vectors for the 18 effective detection features and then optimize them using the spatial and temporal correlation mechanisms. We refer to the first as spatial correlation because we first generated, and then operated, a process relationship forest. In spatial correlation, we updated the current process feature vector by its union with daughter processes. We call the second approach temporal correlation because it combines information from different time windows. Temporal correlation updates the current process feature vector by summing its current feature vector with those in historical time windows.

4.2.1. *Spatial correlation*

Since social bots use the advanced evasion mechanism E_a^1 (exploiting multiple processes), we designed a spatial correlation mechanism to detect it. If a social bot distributes its malicious behaviors into several processes, it will face a critical challenge of robustness, since all the processes must be started, must find each other using some method, and must then communicate with each other. Although social bots can use complicated multiple process models, they must strike a balance between concealment and robustness (Ji et al., 2014). Due to these facts and based on our observations, social bots operate according to two basic multiprocess models – sibling processes or parent and children processes. Other possible models involve various combinations of these two basic models. Therefore, if we can capture the basic models, we can conquer the multiple process evasion mechanism.

In the first model, shown in Fig. 2a, the bot processes are siblings, which belong to a common parent process. These bot processes can communicate with each other directly by process communication methods, such as inter-process communication (IPC), and indirectly through the agent or parent processes. In the second model, shown in Fig. 2b, a bot *Bot_1.exe* acts as the parent process and creates other bot processes, such as *Bot_2.exe*. These bot processes can use a simple communication method – parent and children processes – to communicate with each other.

To combat these communications, we examined the relationships between different processes. They form several precise disjointed trees, also known as a forest. To fully address the special multiple process mechanism of social bots, we iterate each process node from its root to bottom of each process tree. We unite the feature vector of the iterated node with its children nodes to generate a new feature vector. Since the impact of the children process on the iterated node decreases with distance, we use distance decay to present this trend. Distance decay is a geographical term, which describes the effect of distance on spatial interactions. By spatial correlation, we unite the children processes to combat the multiple process mechanisms used by social bots.

$$S_d^i = \alpha_0 V_d^i + \alpha_1\left(\sum_{j\in U_{d+1}} V_{d+1}^j\right) + \cdots + \alpha_n\left(\sum_{j\in U_{d+n}} V_{d+n}^j\right), \quad (2)$$

Suppose we obtain the feature vector of each process and use V_d^i to denote the *i*th process at depth *d*. We use S_d^i to denote the new feature vector and calculate using Equation (2). We use

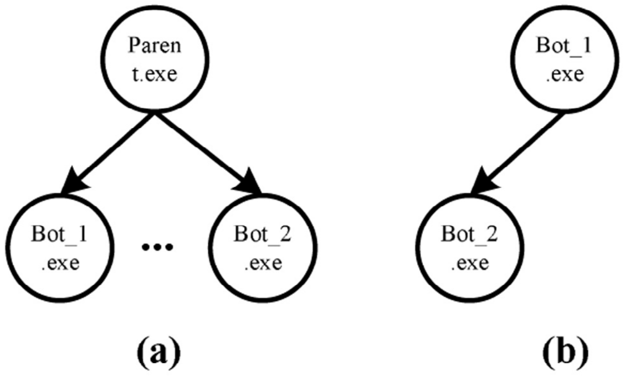

Fig. 2 – Multiple processes mechanisms.

the distance *k* between node *d* and its children nodes to calculate the distance decay, $\alpha_k = a/(k+1)^2$, where *a* is a parameter that balances the decay rate. In Equation (2), *n* is the correlation depth, and U_{d+k} denotes the set of children nodes at depth *k* of node *d*.

After identifying a suspicious process tree from node N_d^i to depth *k*, we identify the specific multiple processes of social bots based on the monitor information. Specifically, we try to extract the processes – which have accessed the same Registry key, file, IP, and domain – and combine them in a group. We do not monitor process communications since it requires a large amount of performance overhead, as proved in Ma et al. (2012). By dividing the original process group into subgroups, we can locate the specific process group of the social bots.

4.2.2. *Temporal correlation*

To detect social bots in real-time, many approaches including ours use time windows. However, time windows are vulnerable to the advanced evasion mechanism, E_a^4, delayed response. To counter this evasion mechanism, we design the temporal correlation mechanism. Our primary heuristic attempt was to the historical information to counteract the flaw. For the current time window, we accumulate the feature vector values of the historical time windows. Based on our observations, the time delay in multiple process bots cannot be significantly high, otherwise the session can be interrupted or cache information lost. Therefore, our temporal correlation mechanism follows two rules: (1) accumulating recent time windows with a digression trend; (2) omitting time windows, that exceed a certain time limitation. Based on the above considerations, we combine the weighted moving average (WMA) and exponentially weighted moving average (EWMA) algorithms.

The moving average is the convolution of the datum points with a fixed weighting function. Specifically, a WMA means the weights that decrease in arithmetical progression. Equation (3) presents a typical WMA with weight *n*, where the latest has weight *n*, the second latest *n* − 1, and so on down to 1. In more general cases, the denominator is always the sum of the individual weights. WMA follows two rules, while the degression in arithmetical progression cannot reflect the real changes in multiple process bots.

$$WMA_M = \frac{nY_M + (n-1)Y_{M-1} + \cdots + Y_{M-n+1}}{n + (n-1) + \cdots + 1} \quad (3)$$

EWMA is a type of infinite impulse response filter that applies weighting factors that decrease exponentially. The weighting factor for each older value decreases exponentially, never reaching zero. Equation (4) presents the typical EWMA algorithm, which can reflect changing trends better than the WMA, as the weights cannot reach zero. Therefore, we combine the two moving average algorithms to represent the weighting changes.

$$S_t = \begin{cases} Y_1, & t = 1 \\ \alpha Y_t + (1-\alpha)S_{t-1}, & t > 1 \end{cases} \quad (4)$$

Since EWMA can represent the current changing weighting trends and WMA can deal with older information, we use

Table 8 – Detection results.

Classifier	FP rate	FN rate	F-measure	Detection rate
J48	0.003	0.072	0.95	0.989
Random tree	0.006	0.058	0.946	0.988
Random forest	0.003	0.047	0.963	0.992
Decision table	0.004	0.185	0.883	0.975

Table 9 – Detection result comparison.

Approach	FP rate	FN rate	F-measure	Detection rate
A	0.103	0.397	0.503	0.863
B	0.097	0.486	0.455	0.858
C	0.105	0.465	0.456	0.854
D	0.003	0.072	0.95	0.989

the EWMA algorithm to accumulate the values in recent time windows and the WMA algorithm to accumulate the values in older time windows. Suppose we use T_1 to denote the time limit of the recent time windows, T_2 to denote the period of time over which the feature vectors are averaged, V_t to denote the feature vector of the t-th time window, and S_t to denote the new feature vector after temporal correlation. Based on Equation (3) and Equation (4), we combine them to obtain a complete temporal correlation algorithm, as presented in Equation (5). In this equation, α is a smoothing factor that is defined as a function of the time interval between two adjacent time windows, $\alpha = 1 - e^{-\frac{t_n - t_{n-1}}{T_1}}$. p denotes the number of time windows used for EWMA, $p = \frac{T_1}{t_n - t_{n-1}} - 1$, and q denotes the number used for WMA, $q = \frac{T_2 - T_1}{t_n - t_{n-1}}$, where $t_n - t_{n-1}$ is the length of the time window.

$$\begin{aligned} S_t &= \alpha V_t + \alpha(1-\alpha) V_{t-1} + \cdots + \alpha(1-\alpha)^p V_{t-p} \\ &\quad + \alpha(1-\alpha)^p \frac{q V_{t-p-1} + (q-1) V_{t-p-2} + \cdots + V_{t-p-q}}{q(q+1)/2} \end{aligned} \tag{5}$$

5. Evaluation

In this section, we evaluate the performance of our detection approach. First, we compare the detection performance with other existing approaches. Then, we evaluate our newly identified features with respect to detecting social botnets. We also validate the effectiveness of the two correlation mechanisms.

5.1. *Detection performance*

We evaluate our approach on the entire collection of data traces described in Section 3.1. Among these traces, there are 31,165 benign instances, and 4042 malicious instances. We conduct our evaluation using four different machine learning classifiers: J48, random tree (RT), random forest (RF), and decision table (DT). For each machine learning classifier, we use the five-fold cross validation method to compute four performance metrics: false positive rate (FPR), false negative rate (FNR), F-measure[2], and detection rate (DR).

Table 8 presents the detection performances of the four classifiers. For the false positive (FP) rate, the rates are all under 1%. In particular, for both J48 and RF, the FP rate is only 0.3%. For the false negative (FN) rate, except for DT, the other three rates are under 8%. Decision Table has a higher FN rate of 18.5%, which is also fairly good. From these FP and FN rates, we confirm that our approach can accurately classify benign instances and social bots. For the F-measure, J48, RT, and RF are all about 0.95, while DT has a lower value of 0.883, which is also fairly good. For the DR, all the classifiers achieve better than 97%, and RF reaches 99.2%. From the above analysis, we confirm that our approach can achieve a high detection rate according to different classifiers. Although DT has the lowest detection result, its FP rate is significantly high and its final detection rate is fairly good. The classifier RF achieve the best result of the evaluation metrics employed.

To clearly represent the detection results, we next compare our approach with other existing social bot detection approaches. We label three existing social bot detection approaches as A, B, C (as before, where work of Kartaltepe et al. (2010) is A, that of Chu et al. (2013) is B, that of Burghouwt et al. (2011) is C, and we labeled our work D.) Since J48 has a moderate detection performance among the four classifiers, we chose it to represent our approach for comparison with others.

We also use the four evaluation metrics – FP rate, FN rate, F-measure, and Detection Rate – to present our detection results. From Table 9, we can see that the FP rate of our approach is the lowest, below 1%, while the FP rates of approaches A, B, and C are approximately 10%. This low FP rate indicates that our approach can classify benign instances more accurately than existing approaches. For the FN rate, our approach also outperforms existing approaches. Its approach has an FN rate of about 7.2%, while the FN rate of the best existing approach A is 39.7%. The FN rates of the other two approaches, B and C, are approximately 47%. As discussed above, the FN rates of the existing approaches are significantly higher since the evolving social bots utilize more advanced evasion mechanisms to evade their detection rules. With its newly identified features and correlation mechanisms, our approach can identify social bots with reasonable accuracy. With respect to the F-measure values. approaches A, B, and C are all below 0.51, while those of our approach reaches as high as 0.95. For the detection rate, approaches A, B, and C are all about 86%, which is fairly good. However, our approach achieve the highest detection rate of 98.9%, which is highly significant.

5.2. *Feature validation*

In this section, we present the validation of our newly identified features. First, we compare detection results with and without our newly identified features. Then, we present the measurements of the specific features.

5.2.1.1. *Feature set comparison.* Fig. 3 compares the detection results with and without our newly identified features. Under each evaluation metric, we show the results

[2] F-measure is a measure with the consideration of both precision and recall.

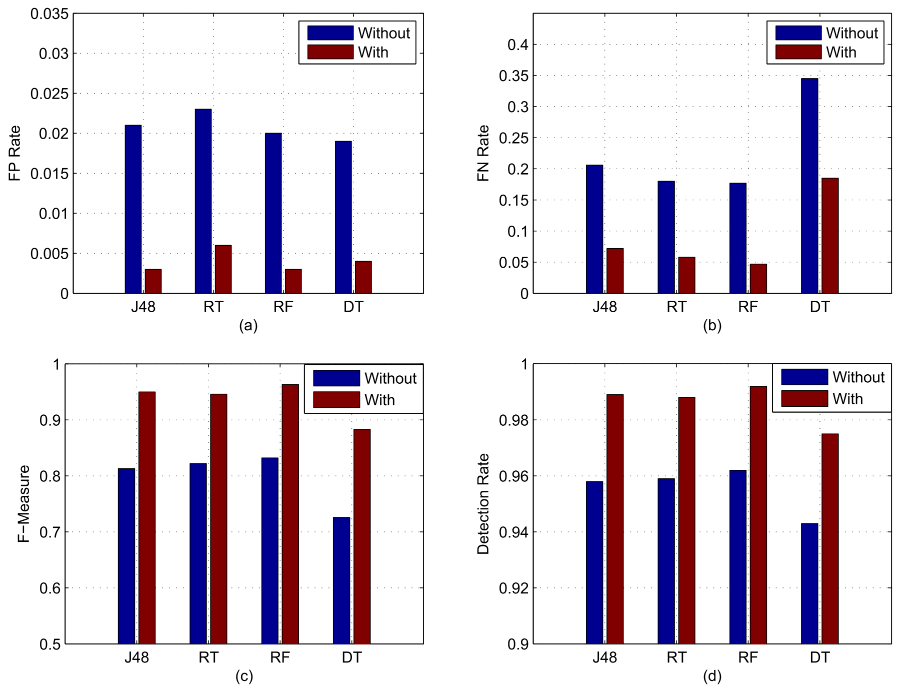

Fig. 3 – Detection results comparison with and without our newly identified features.

of all four classifiers with and without our newly identified features.

As shown in Fig. 3a, all the classifiers without our features have a 2% FP rate, which is fairly good. However, the FP rate of all the classifiers can drop to about 0.5% when using our newly identified features, which is significant. For the classifiers J48 and RF, in particular, the FP rates are about 0.3%. Based on the evaluation metric FP rate results, we can see that our newly identified features can further improve the classification accuracy of benign instances. As shown in Fig. 3b, except for the classifier Decision Table (DT), the other three classifiers reach an FN rate of about 20% without our features, while the FN rate is further decreased to about 5% by incorporating our newly identified features. This 15% improvement of in the FN rate is significant, which means that our newly identified features can further improve the identification accuracy of evolving social bots. Although the Decision Table (DT) has a high FN rate of about 35%, it can also showed improvements of 17–18%, which is fairly good.

As shown in Fig. 3c, the F-measure values of J48, RT and RF without our features are about 0.8, while DT is about 0.72. However, the F-measure values of the first three classifiers improve by 0.15 to about 0.95, which is significant. Even the DT F-measure value reaches about 0.9, representing an improvement of about 0.18. The improvements in F-measure values indicate that our newly identified features increase the efficiency and effectiveness of detecting social bots. Similar conclusions can also be made regarding the detection rate (DR). Without our newly identified features, the first three classifiers have a detection rate of about 96%, and Decision Table (DT) of about 94%. These detection rates are fairly good. However, with our features, the first three classifiers improve by 3% to about 99%, and the DT improves from about 4% to about 98%, both of which represent significant improvements.

5.2.1.2. *Feature rank.* Using all the features, our approach can achieve a significant detection result for social botnet. However, different features play different detection roles. To show the weights of each feature, we rank them using three prevalent feature evaluation methods: the Chi-square test, information gain, and gain ratio. For each method, we rank the features, as shown in Table 10. The results show that our newly identified features are effective in detecting social bots. In particular, the pre-defined host features (F_1 to F_4) are nearly ranked almost highest by all three methods. Features F_2, F_1, and F_4 are ranked as the top three by the different methods, while F_3 is ranked last by all. C&C connection features (F_5 to F_{10}) are also highly ranked in different methods. In addition, the failure-based features (e.g., F_{17} the number of failed OSN domains) are also very effective in discovering social bots. From this rank analysis, we observe that a number of existing features remain effective in detecting social bots, including the pre-defined host features and C&C connection features. The combination of existing features and our newly identified features has yielded an approach that can effectively detect social bots.

From the feature ranking table, we can also observe that some features have low rank values (e.g., F_3 was ranked last and F_{12}(*) second to last). This observation raise the question of whether features with low rank values are unimportant and can therefore be excluded. To answer these questions, we performed further experiments in which we removed the low-ranked features, and assumed that the top ten highly ranked features should not be removed. Based on gain ratio ranking, we performed another eight rounds in which we progressively removed

Table 10 – Feature ranking.

Index	Feature	Chi-square (rank)	Information gain (rank)	Gain ratio (rank)
F_1	Modifying bootstrap list of system	2	2	2
F_2	Stealing sensitive information	1	1	1
F_3	Process injection	18	18	18
F_4(*)	Checking Internet cookies	3	3	3
F_5	Number of unique IPs contacted	6	4	6
F_6	Number of unique domains queried	4	5	5
F_7	Number of visited OSN IPs	8	8	8
F_8(*)	Number of visited OSN accounts	7	7	7
F_9(*)	Number of visited OSN user messages	10	10	10
F_{10}(*)	Number of visited OSN user pictures	9	9	9
F_{11}	Number of opened ports	13	13	13
F_{12}(*)	Number of uploaded OSN messages	17	17	17
F_{13}(*)	Number of uploaded OSN pictures	16	16	16
F_{14}	Number of failed IPs	15	15	15
F_{15}	Number of failed domains	14	14	14
F_{16}(*)	Number of failed OSN queries	12	12	12
F_{17}(*)	Number of failed OSN domains	4	6	4
F_{18}(*)	Number of failed visited OSN accounts	11	11	11

*represents our newly identified features.

the low-ranked features. We used the same evaluation metrics and machine learning classifiers as in the other experiments. The detection results are shown in Fig. 4. In each figure, the number on the x axis represents the number of features used, meaning that we removed the last 18 – *n* features.

In the FP rate shown in Fig. 4a, the difference between 10 and 18 was small for the classifiers J48, RF, and DT, while the RT classifier showed a bigger difference. Overall, the FP rate gradually declined. A similar observation was made for the FN rate in Fig. 4b. As indicated by the F-measure values and the detection rate in Fig 4c and d, the overall trend rose gradually. The comparison of results using 10 and 18 features showed only a small difference. From these experimental results, we nearly concluded that the top ten features played a dominant role in detecting social botnets, while the eight lowest features yielded only a small impact. However, before considering this conclusion as decisive, note that social botnet traces may display bias, such as feature F_{18}(*), the number of failed visited OSN accounts. In our traces, the botmaster's accounts were almost always accessible. We discuss this bias in more

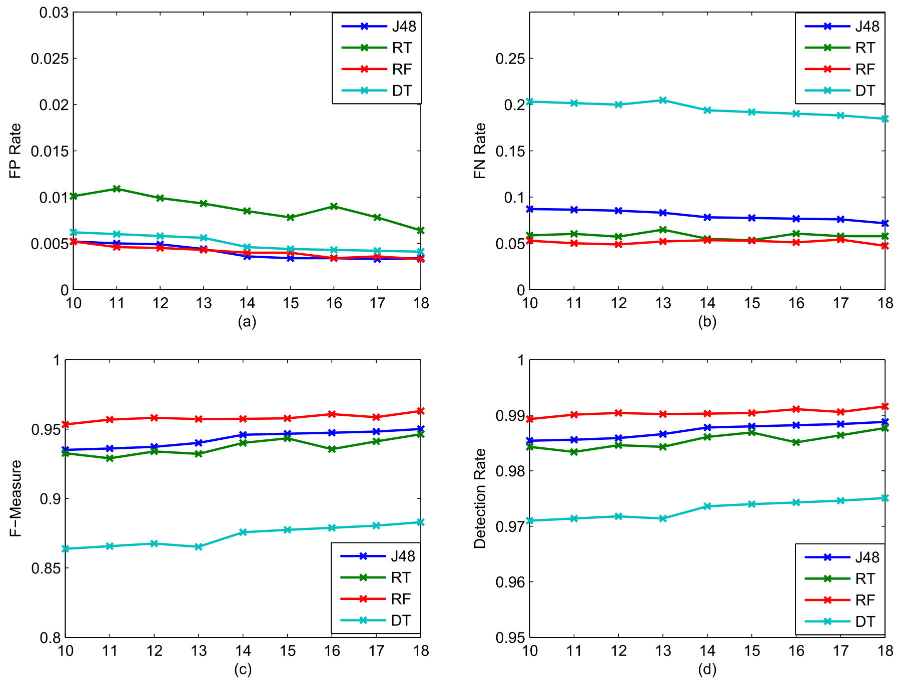

Fig. 4 – Detection result with low rank features removed.

detail in Section 5.5. In this study, we attempted to combat the evasion mechanisms of evolving social botnets. Although the low-ranked features may play a small role in detecting existing social botnets, they may be significant in capturing evolving social botnets. Taking feature F_{13}(*), the number of uploaded OSN pictures, as an example, existing social botnets encounter problems in using pictures to hide information. However, possible solutions have been discussed, with respect to Stegobot and Facebot (Nagaraja et al., 2011; Verkamp et al.). If we remove this feature, our approach may be unable to combat emerging social botnets that use pictures to hide information.

Based on these considerations, we intend to preserve all features in our approach. However, these observations have led us to decide to undertake further research on these features in future work, with the aim of removing old-fashioned or ineffective features and introducing more effective ones. We would also like to try adaptive methods to enable our approach to automatically generate effective features and remove ineffective ones.

5.3. *Correlation mechanisms validation*

To present the effectiveness of our designed correlation mechanisms, we compare the detection results of four situations, without correlation, spatial only, temporal only, spatial and temporal correlation. We evaluate the correlation mechanisms with appropriate parameters.

As shown in Fig. 5a, the FP rates of all classifiers without correlation mechanisms are about 2% which is fairly good. With temporal correlation only, there is a slight decrease in the FP rates of J48 and DT, while there is a slight increase in the FP rates of RT and RF. This indicates that temporal correlation has an unstable influence on the classification of benign instances. With spatial correlation only, the FP rates of all four classifiers significantly decrease. Although RT has a decrease of only about 0.5%, the other three have a significant decrease of about 1.5%. This means that spatial correlation can effectively improve the classification of benign instances. With both correlation mechanisms, the FP rates of J48, and RF decrease compared with those of spatial correlation, while DT has a slight increase from that of spatial correlation and RT has a significant decrease. From this analysis of FP rates, we confirm that the two correlation mechanisms are associated with a significant improvement in the detection of benign instances.

From Fig. 5b, we observe that the FN rates of all clasifiers without correlation mechanisms are about 17%. With temporal correlation, the FN rates of all the classifiers have a slightly increase, indicating that temporal correlation cannot improve the detection of social bots. However, we note that the temporal correlation mechanism accumulates the features of related time windows. Some social bot processes have a short life span; therefore, the featab: collected tracetures are not accumulated to have a better detection result. In this situation, temporal correlation can result in some false negatives. With spatial correlation, the FN rates of J48, RT, and RF have a slightly decrease, while the FN rate of DT increases slightly. With both temporal and spatial correlation mechanisms, the FN rates of J48, RT, and RF decrease significantly by about 10%, while the FN rate of DT increases slightly. Although the DT classifier shows a slight increase into the FN rate with both correlation mechanisms, the other three achieve a significant decrease into the FN rates.

From Fig. 5c and d, although the temporal correlation mechanism causes a slight decrease in the F-measure and detection rate, the combination of temporal and spatial correlation mechanisms results in a significant increase in F-measures and detection rates.

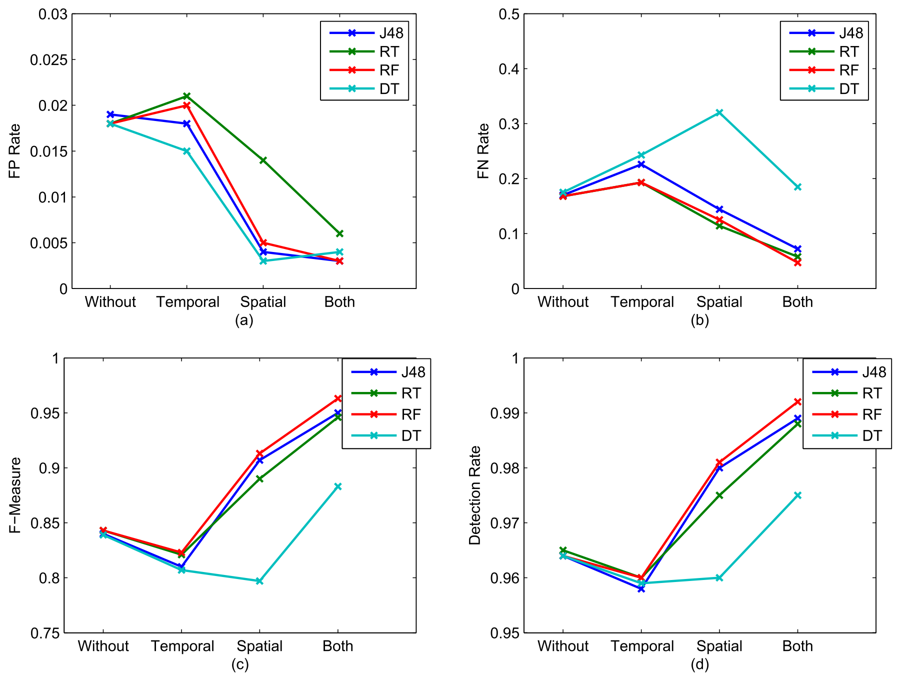

Fig. 5 – Detection results comparison of correlation mechanisms.

5.4. *Detecting common botnets*

We evaluated the success of our approach in detecting both social and common botnets. We used the common botnet dataset in BotCatch (Ji et al., 2015) to evaluate detection performance, which comprises 636 bot samples, including 47 different types of bots and 150 benign samples. BotCatch is an effective multi-feedback host-side botnet detection approach that uses a combination of signature and behavior.

The support vector machine (SVM) correlation combines behavioral and signature results to yield a comprehensive detection result, which is used as the benchmark in Ji et al. (2015). We ran comparisons to test whether our approach was effective in detecting common botnets. To show the different roles played by each feature played in detecting common and social botnets, we describe below the feature rankings of our approach in detecting common botnets.

5.4.1.1. *Detection performance*. The detection results are shown in Fig. 6. In each figure, the x axis indicates the four classifiers, while Bench_SVM and Bench_BotCatch are benchmark approaches that are not related to the classifiers. As shown in Fig. 6a, the FP rates of our approach were between 10% and 15%. Among the four classifiers, RF achieved the lowest FP rate, at about 11.4%, while DT had the highest FP rate, at about 14.8%. When compared with the 0% FP rate of BotCatch, our approach did not perform well. However, when compared with the 8.7% FP rate of SVM correlation, our approach was just 2.7% higher. We consider the FP rate of our approach to be acceptable because it accurately classified most benign samples. The FN rates of our approach in detecting common botnets were between 11% and 6%, as shown in Fig. 6b. More specifically, RF recorded the lowest FN rate, at about 6.4%, and RT had the highest FN rate, at about 10.6%. The benchmark approach SVM correlation had a 2% FN rate, and BotCatch had 3.6%. The FN rate of our approach was only 2.8% higher than that of BotCatch, and 4.4% higher than that of the SVM correlation. The low FN rate suggests that most common bots can be identified by our approach. As shown in Fig. 6c, the classifier RF had the highest F-measure value for our approach, at about 0.958, which is only 0.022 lower than that of the SVM correlation and 0.024 lower than that of BotCatch. Moreover, the classifier RF had the highest detection rate among the four classifiers, as shown in Fig. 6d; it achieves about 92.9%, which was only 3.8% lower than that of the SVM correlation and 4.2% lower than that of BotCatch. These detection results show that our approach performed less well than either BotCatch or the SVM correlation in detecting common botnets, but with classifier RF, it achieved an 11.4% FP rate, a 6.4% FN rate, a 0.958 F-measure value, and a 92.9% detection rate. Moreover, note that the two benchmark approaches used both signature and behavior detection results, whereas our approach was solely behavior based. In light of this, our approach achieved acceptable results in the detection of common botnets.

5.4.1.2. *Feature rank*. Although our approach can detect both social botnets and common botnets, each feature played a different detection role. Table 11 ranks the features in our approach with respect to detecting common botnets using the same feature evaluation methods used in Section 5.2. Since the results of the three methods were almost identical, we use information gain to explore the results.

The table suggests two interesting observations. First, only the top six features (F_{11}, F_5, F_2, F_6, F_{15}, and F_1) have entropy values, while all other features had values of 0, which means that only the top six features were useful in detecting common botnet datasets. This shows that features play different roles when detecting common botnets and social botnets. Another inter-

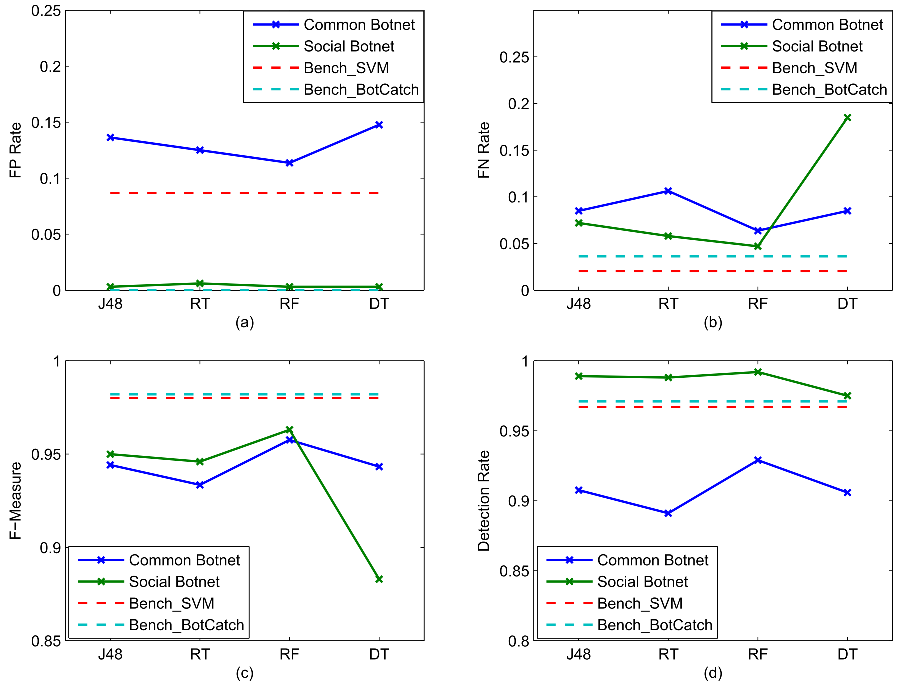

Fig. 6 – Detection results for common botnet.

Table 11 – Feature ranking for detecting common botnet.

Index	Feature	Chi-square (rank)	Information gain (rank)	Gain ratio (rank)
F_1	Modifying bootstrap list of system	6 (15.403)	6 (0.0219)	6 (0.0265)
F_2	Stealing sensitive information	3 (313.464)	3 (0.2748)	3 (0.1693)
F_3	Process injection	9 (0)	9 (0)	9 (0)
F_4(*)	Checking Internet cookies	7 (0)	7 (0)	7 (0)
F_5	Number of unique IPs contacted	2 (486.378)	2 (0.4479)	1 (0.5102)
F_6	Number of unique domains queried	4 (48.66)	4 (0.0607)	5 (0.062)
F_7	Number of visited OSN IPs	10 (0)	10 (0)	10 (0)
F_8(*)	Number of visited OSN accounts	8 (0)	8 (0)	8 (0)
F_9(*)	Number of visited OSN user messages	18 (0)	18 (0)	18 (0)
F_{10}(*)	Number of visited OSN user pictures	16 (0)	16 (0)	16 (0)
F_{11}	Number of opened ports	1 (524.291)	1 (0.4968)	2 (0.3864)
F_{12}(*)	Number of uploaded OSN messages	17 (0)	17 (0)	17 (0)
F_{13}(*)	Number of uploaded OSN pictures	15 (0)	15 (0)	15 (0)
F_{14}	Number of failed IPs	13 (0)	13 (0)	13 (0)
F_{15}	Number of failed domains	5 (28.67)	5 (0.0562)	4 (0.0744)
F_{16}(*)	Number of failed OSN queries	14 (0)	14 (0)	14 (0)
F_{17}(*)	Number of failed OSN domains	12 (0)	12 (0)	12 (0)
F_{18}(*)	Number of failed visited OSN accounts	11 (0)	11 (0)	11 (0)

*represents our newly identified features.

esting observation is that four of the top six features for detecting social botnets 4 (F_5, F_2, F_6, and F_1) are also top ranked for detecting common botnets. In addition, five of the six features were categorized as lifecycle-based, and only one as failure-based. F_2 and F_1 belong to botnet lifecycle step 2 (predefined host features), while F_5 and F_6 belong to botnet lifecycle step 3 (C&C connection features), and F_{11} belong to step 4 (command execution features). This suggests that while botnets can sacrifice benefits to evade multiple features, they cannot evade all the features in an entire lifecycle. In other words, although botnets evolve with different evasion mechanisms, they still cannot rid themselves of robust or critical features. This conclusion precisely supports the motivation for our work.

5.5. *Summary and discussion*

The evaluation results suggest that our approach has better social botnet detection performance than existing approaches. However, it remains to be seen whether the new features and correlation mechanisms of our approach can overcome the disadvantages identified in the studies (Burghouwt et al., 2011; Chu et al., 2013; Kartaltepe et al., 2010). At the general level, all three existing approaches will benefit from the new features and correlation mechanisms. Our life-cycle based features can extend their detection model to cover more social botnet evasion mechanisms, and the failure-based features can assist them to capture malicious attempts before a social bot becomes fully active. Spatial correlation mechanisms can help them to address the challenges of multiple process evasion mechanisms, and temporal correlation can help them to fight against delayed response. However, these three approaches have other disadvantages, as described in Section 3.3.2. While our new features and correlation mechanisms would improve their approaches in some areas, further optimization is required to improve the detection results for social botnets. Approach A (Kartaltepe et al., 2010) can be improved by adding two further rules and adding our new features and correlation mechanisms. First, the detection attributes that are difficult to quantify should be removed, such as encoded text processing (P_{etp}), and second, the excessively deterministic detection model should be relaxed, for example by adding more *or* ($\vee$), or decreasing the number of *and* ($\wedge$). Approach B (Chu et al., 2013) mainly focuses on detecting blog bots, but to better detect social bots, the browser level monitor should be updated to system level, and more rules should be added to distinguish between social bots and automatic programs. Our correlation mechanisms can help approach C to solve the problem of multiple process and delayed response evasion mechanisms. Approach C (Burghouwt et al., 2011) should also solve the problem of different time intervals between network traffic and human operation under different situations, such as different computers, network situations, or human users.

We evaluate most of the existing social bots, and reproduced some of them. This is necessary because we were only able to acquire the source codes or builders of some selected social bots, and not all of them. To better empirically evaluate existing social bots, an analysis of most of the existing social bots is necessary. Therefore, we reproduce existing social bots based on other researchers' descriptions or analyses. Even though we lacked a sufficient number of real social bots, we were able to re-create many of them to enable accurate analysis. One of our key contributions is the collection of social bots, and we intend to continue to collect them to test our approaches.

A similar situation occurs when comparing other detection approaches. We reproduced two of the three approaches used for comparison based on their original papers. We acknowledge that some bias may exist in the reproductions.

In approach A (Kartaltepe et al., 2010), among the eight detection rules mentioned in Section 3.3, several specific points are insufficiently clear. For example, in encoded text processing (P_{etp}), the specific monitored encoded methods are not presented. Based on other papers and technical reports, we tried our best to reproduce the approach using as much information as we could obtain. Approach B designed a webpage-embedded logger to collect user input activities (Chu et al., 2013). However, a social botnet can evade the logger using hidden browsers or the APIs of browsers (Ji et al., 2014). To make information collection more accurate, we designed a system level keyboard and mouse monitor to collect all user input activities and then extracted those related to webpages. Collecting information in this way differs somewhat from the original approach and consumes more resources. However, we believe that it is acceptable to sacrifice a reasonable amount of resources to collect more accurate information.

As discussed in the work of García – an empirical comparison of botnet detection approaches (Garca et al., 2014) – it is difficult to compare detection approaches, since the datasets are private and the descriptions of the methods are incomplete. Given that we used both original detection approaches of some and re-created others based on original papers, the comparison between our work and that of previous studies is still accurate and convincing. Similar situations occurred in dataset collection. In our future work, we will try to use more real detection approaches to achieve more accurate comparison.

In our data collection phase, there are two limitations. First, we collect the data using a virtual machine. Some botnets can detect whether they are running in a virtual machine before performing malicious activities. This is a common challenge to all virtual machine based detection approaches. However, to the best of our knowledge, this kind of evasion mechanism has not yet been identified in social bots. Therefore, this limitation does not currently have any impact on the detection results of our approach. Should the migration of this evasion mechanism from other bots to social bots occur, we will try to improve the data collection phase in our future work. Another limitation is that we used monitor tools to collect data. These tools cease monitoring for several seconds while saving the monitored data. During this interval, the data collected are not accurate. However, compared to the entire time windows (20 minutes in our experiment), the lose of data over several seconds does not change the results. To make our detection approach more accurate, we will try to collect more accurate data in our future work.

In future work, we plan to analyze more social bot samples, and identify different kinds of evasion mechanisms. Based on the new findings of our study, we would like to design more effective and efficient detection features. We also plan to broaden our detection location from host only to correlate the host and network. By analyzing the abnormal network flow changes caused by social bots, we may observe some intrinsic social bots behaviors. We can also combine our work with OSN spammer detecting approaches. By correlating their host behaviors and actual OSN behaviors, we may discover some additional features to effectively combat social bots.

6. Conclusion

In this paper, we present an empirical evaluation on evolving social botnet evasion mechanisms and design a new approach to combat them. First, we analyze the evasion mechanisms utilized by social bots to evade existing detection approaches. We validate these mechanisms on three state-of-the-art detection approaches based on our collected traces and theoretically analyze the reasons for their low detection result. We then design a new approach to combat social bots, involving nine newly identified features and nine existing features, classified as either lifecycle- and failure-based. We also design two correlation mechanisms – spatial and temporal correlations. Finally, based on the evaluation results, we conclude that our approach performs much better than existing detection approaches. Our approach with the classifier RF obtained about 0.3% FP rate, 4.7% FN rate, 0.963 F-measure value and 99.2% detection rate. We verified the effectiveness of our newly identified features and correlation mechanism in the experiment. In addition to the detection of social bots, we also evaluate our approach with respect to the detection of common bots.

Acknowledgment

Social botnet samples are, to a great degree, provided by other researchers. We are most thankful for their generosity in sharing their samples (and their advice) with us. We thank Pieter Burghouwt for sharing the source code of his team's approach with us. This work is supported by the National Natural Science Foundation of China under Grant No. 61170265 and Grant No. 61472162.

REFERENCES

Abokhodair N, Yoo D, McDonald DW. Dissecting a social botnet: growth, content and influence in twitter, in: Proceedings of the 18th ACM conference on computer supported cooperative work & social computing, ACM, 2015, pp. 839–851.
Ahn G-J, Shehab M, Squicciarini A. Security and privacy in social networks. IEEE Internet Comput 2011;15(3):10–12.
Al-Hammadi Y., Aickelin U., Detecting botnets through log correlation, arXiv preprint arXiv:1001.2665.
Alotibi G, Li F, Clarke N, Furnell S. Behavioral-based feature abstraction from network traffic, in: ICCWS 2015-The proceedings of the 10th international conference on cyber warfare and security, Academic Conferences Limited, 2015, p. 1.
Anon I heart quotes, <https://github.com/robertodecurnex/i_heart_quotes>; 2015a [accessed 03.15].
Anon Two million stolen facebook, twitter, yahoo, adp passwords found on pony botnet server, <http://www.zdnet.com/two-million-stolen-facebook-twitter-yahoo-adp-passwords-found-on-pony-botnet-server-7000023915/>; 2015c [accessed 03.15].
Anon "pony" botnet steals bitcoins, digital currencies: trustwave, <http://www.reuters.com/article/2014/02/24/us-bitcoin-security-idUSBREA1N1JO20140224>; 2015d [accessed 03.15].
Avery J, Gutierrez C, Wood P, Corte RD, Modelo-Howard JFG, Berndt B, et al., Snipe: signature generation for phishing emails, in: Proceedings of the 16th annual information security symposium, CERIAS-Purdue University, 2015, p. 14.
Baltazar J, Costoya J, Flores R. The real face of koobface: the largest web 2.0 botnet explained. Trend Micro Research 2009;5(9):10.
Boshmaf Y, Muslukhov I, Beznosov K, Ripeanu M. Key challenges in defending against malicious socialbots, in: Proceedings of the 5th USENIX conference on large-scale exploits and emergent threats, USENIX Association, 2012, pp. 12–12.
Boshmaf Y, Muslukhov I, Beznosov K, Ripeanu M. Design and analysis of a social botnet. Comput Netw 2013;57(2):556–78.

Boshmaf Y, Logothetis D, Siganos G, Lería J, Lorenzo J, Ripeanu M, et al., Íntegro: leveraging victim prediction for robust fake account detection in osns, in: Proc. of NDSS, 2015.
Brito F, Petiz I, Salvador P, Nogueira A, Rocha E. Detecting social-network bots based on multiscale behavioral analysis, in: SECURWARE 2013, The seventh international conference on emerging security information, systems and technologies, 2013, pp. 81–85.
Burghouwt P, Spruit M, Sips H. Towards detection of botnet communication through social media by monitoring user activity, in: Proceedings of the 7th international conference on information systems security, ICISS'11, 2011, pp. 131–143.
Burghouwt P, Spruit M, Sips H. Detection of covert botnet command and control channels by causal analysis of traffic flows. In: Cyberspace Safety and Security. Springer; 2013. p. 117–31.
Cao J, Li Q, Ji Y, He Y, Guo D. Detection of forwarding-based malicious urls in online social networks. Int J Parallel Program 2016;44(1):163–80.
Cao Q, Yang X, Yu J, Palow C. Uncovering large groups of active malicious accounts in online social networks, in: Proceedings of the 2014 ACM SIGSAC conference on computer and communications security, ACM, 2014, pp. 477–488.
Chandramouli R. Analysis of network segmentation techniques in cloud data centers, in: Proceedings of the international conference on grid computing and applications (GCA), The Steering Committee of The World Congress in Computer Science, Computer Engineering and Applied Computing (WorldComp), 2015, p. 64.
Chu Z, Gianvecchio S, Wang H, Jajodia S. Detecting automation of twitter accounts: are you a human, bot, or cyborg? IEEE Trans Dependable Secure Comput 2012;9(6):811–24.
Chu Z, Gianvecchio S, Koehl A, Wang H, Jajodia S. Blog or block: detecting blog bots through behavioral biometrics. Comput Netw 2013;57(3):634–46.
Dietrich CJ, Rossow C, Pohlmann N. Cocospot: clustering and recognizing botnet command and control channels using traffic analysis. Comput Netw 2013;57(2):475–86.
Douligeris C, Mitrokotsa A. Ddos attacks and defense mechanisms: classification and state-of-the-art. Comput Netw 2004;44(5):643–66.
Edwards C, Edwards A, Spence PR, Shelton AK. Is that a bot running the social media feed? testing the differences in perceptions of communication quality for a human agent and a bot agent on twitter. Comput Human Behav 2014;33:372–6.
Elyashar A, Fire M, Kagan D, Elovici Y. Homing socialbots: intrusion on a specific organization's employee using socialbots, in: Proceedings of the 2013 IEEE/ACM international conference on advances in social networks analysis and mining, ACM, 2013, pp. 1358–1365.
Faghani MR, Nguyen UT. Socellbot: a new botnet design to infect smartphones via online social networking, in: Electrical & Computer Engineering (CCECE), 2012 25th IEEE Canadian conference on, IEEE, 2012, pp. 1–5.
Ferrara E., Varol O., Davis C., Menczer F., Flammini A., The rise of social bots, arXiv preprint arXiv:1407.5225.
Freitas C.A., Benevenuto F., Ghosh S., Veloso A., Reverse engineering socialbot infiltration strategies in twitter, arXiv preprint arXiv:1405.4927, 2015.
Garca S, Grill M, Stiborek J, Zunino A. An empirical comparison of botnet detection methods. Comput Secur 2014;45:100–23.
Garfinkel SL. The cybersecurity risk. Commun ACM 2012;55(6):29–32.
Gritzalis D, Kandias M, Stavrou V, Mitrou L. History of information: the case of privacy and security in social media, in: Proc. of the history of information conference, 2014, pp. 283–310.
Heikkila FM. Data privacy in the law firm: how to protect client data. MICH. BJ 2009;88:33–36.
Hwang T, Pearce I, Nanis M. Socialbots: voices from the fronts. ACM Int 2012;19(2):38–45.
Ji Y, He Y, Jiang X, Li Q. Towards social botnet behavior detecting in the end host, in: 20th IEEE international conference on parallel and distributed systems, ICPADS 2014, Hsinchu, Taiwan, December 16–19, 2014, 2014, pp. 320–327.
Ji Y, He Y, Zhu D, Li Q, Guo D. A multiprocess mechanism of evading behavior-based bot detection approaches, in: information security practice and experience – 10th international conference, ISPEC 2014, Fuzhou, China, May 5–8, 2014. Proceedings, 2014, pp. 75–89.
Ji Y, Li Q, He Y, Guo D. Botcatch: leveraging signature and behavior for bot detection. Secur Commun Netw 2015;8(6):952–69.
Karlesky M, Sae-Bae N, Isbister K, Memon N. Who you are by way of what you are: behavioral biometric approaches to authentication. Symposium on Usable Privacy and Security (SOUPS); 2014.
Kartaltepe EJ, Morales JA, Xu S, Sandhu R. Social network-based botnet command-and-control: emerging threats and countermeasures, in: Proceedings of the 8th international conference on applied cryptography and network security, ACNS'10, 2010, pp. 511–528.
Kolbitsch C, Comparetti PM, Kruegel C, Kirda E, Zhou X-Y, Wang X. Effective and efficient malware detection at the end host, in: USENIX security symposium, 2009, pp. 351–366.
Lalas E, Papathanasiou A, Lambrinoudakis C. Privacy and traceability in social networking sites, in: Informatics (PCI), 2012 16th Panhellenic conference on, IEEE, 2012, pp. 127–132.
Li B, He J, Huang J, Shi YQ. A survey on image steganography and steganalysis. JIHMSP 2011;2(2):142–72.
Ma W, Duan P, Liu S, Gu G, Liu J-C. Shadow attacks: automatically evading system-call-behavior based malware detection. J Comput Virol 2012;8(1–2):1–13.
Maybury M, Chase P, Cheikes B, Brackney D, Matzner S, Hetherington T, et al., Analysis and detection of malicious insiders, Tech. rep., DTIC Document; 2005.
McDonald J, Yasinsac A, Thompson W. A survey on mobile agent security, Tech. rep., Technical report, TR-050329, Dept. of Computer Science, Florida State University. Available from: <http://www.cs.fsu.edu/research/reports/TR-050329.pdf>; 2005.
Moyano F, Fernandez-Gago C, Lopez J. A framework for enabling trust requirements in social cloud applications. Requirements Engineering 2013;18(4):321–41.
Naessens V, Mihaylov M, de Jong S, Verbeeck K, Nowé A. Carebook: assisting elderly people by social networking, in: Proceedings of the 1st international conference on interdisciplinary research on technology, education and communication (ITEC), 2010.
Nagaraja S, Houmansadr A, Piyawongwisal P, Singh V, Agarwal P, Borisov N. Stegobot: a covert social network botnet. In: Information hiding. Springer; 2011. p. 299–313.
Nappa A, Xu Z, Rafique MZ, Caballero J, Gu G. Cyberprobe: towards internet-scale active detection of malicious servers, in: 2014 Network and Distributed System Security (NDSS) symposium; 2014.
Natarajan V, Sheen S, Anitha R. Detection of stegobot: a covert social network botnet, in: Proceedings of the first international conference on security of internet of things, ACM, 2012, pp. 36–41.
Nazario J. Twitter based botnet command and control, <http://asert.arbornetworks.com/2009/08/twitter-based-botnet-command-channel>; 2009 [accessed 03.15].
Neugschwandtner M, Comparetti PM, Platzer C. Detecting malware's failover c&c strategies with squeeze, in:

Proceedings of the 27th annual computer security applications conference, ACM, 2011, pp. 21–30.
Park Y, Reeves DS. Identification of bot commands by run-time execution monitoring, in: Computer security applications conference, 2009. ACSAC'09. Annual, IEEE, 2009, pp. 321–330.
Ragsdale D, Lathrop S, Dodge R. Enhancing information warfare education through the use of virtual and isolated networks. J Inf Warf 2003;2(3):53–65.
Safa NS, Sookhak M, Von Solms R, Furnell S, Ghani NA, Herawan T. Information security conscious care behaviour formation in organizations. Comput Secur 2015;53:65–78.
Schrittwieser S, Frühwirt P, Kieseberg P, Leithner M, Mulazzani M, Huber M, et al., Guess who's texting you? Evaluating the security of smartphone messaging applications, in: NDSS, 2012.
Shin S, Xu Z, Gu G. Effort: Efficient and effective bot malware detection, in: INFOCOM, 2012 Proceedings IEEE, IEEE, 2012, pp. 2846–2850.
Silva SS, Silva RM, Pinto RC, Salles RM. Botnets: a survey. Comput Netw 2013;57(2):378–403.
Singh A. Social networking for botnet command and control, 2012, Master's Projects. Paper 247.
Spafford EH. Privacy and security recalling malware milestones. Commun ACM 2010;53(8):35.
Stein T, Chen E, Mangla K. Facebook immune system, in: Proceedings of the 4th workshop on social network systems, ACM, 2011, p. 8.
Suen T, Yasinsac A. Ad hoc network security: peer identification and authentication using signal properties, in: Information assurance workshop, 2005. IAW'05. Proceedings from the sixth annual IEEE SMC, IEEE, 2005, pp. 432–433.
Tan E, Guo L, Chen S, Zhang X, Zhao Y. Spammer behavior analysis and detection in user generated content on social networks, in: Distributed computing systems (ICDCS), 2012 IEEE 32nd international conference on, IEEE, 2012, pp. 305–314.
Verkamp J.-P., Malshe P., Gupta M., Dunn C.W., Facebot: an undiscoverable botnet based on treasure hunting social networks.
Wagner C, Mitter S, Körner C, Strohmaier M. When social bots attack: modeling susceptibility of users in online social networks, in: Proceedings of the WWW, Vol. 12, 2012.
Wang D, Navathe SB, Liu L, Irani D, Tamersoy A, Pu C. Click traffic analysis of short url spam on twitter, in: collaborative computing: networking, Applications and worksharing (Collaboratecom), 2013 9th international conference on, IEEE, 2013, pp. 250–259.
Yang C, Harkreader R, Gu G. Empirical evaluation and new design for fighting evolving twitter spammers. IEEE T Inf Foren Sec 2013;8(8):1280–93.
Yang Z, Wilson C, Wang X, Gao T, Zhao BY, Dai Y. Uncovering social network sybils in the wild. ACM Trans Knowl Discov Data 2014;8(1):2.
Yu L.L., Asur S., Huberman B.A., Dynamics of trends and attention in Chinese social media, arXiv preprint arXiv:1312.0649.
Zeng Y. On detection of current and next-generation botnets [Ph.D. thesis], The University of Michigan; 2012.
Zhang J, Zhang R, Zhang Y, Yan G. On the impact of social botnets for spam distribution and digital-influence manipulation, in: Communications and Network Security (CNS), 2013 IEEE conference on, IEEE, 2013, pp. 46–54.
Zhang Z, Naït-Abdesselam F, Lin X, Ho P-H. A model-based semi-quantitative approach for evaluating security of enterprise networks, in: Proceedings of the 2008 ACM symposium on applied computing, ACM, 2008, pp. 1069–1074.
Zhao Z, Ahn G-J, Hu H. Examining social dynamics for countering botnet attacks, in: Global telecommunications conference (GLOBECOM 2011), 2011 IEEE, IEEE, 2011, pp. 1–5.

Yuede Ji received his BEng degree in software engineering from Jilin University, China in 2012. Currently, he is an MSc candidate in the Computer Science Department at Jilin University, China. His main research focuses on the detection of botnet.

Yukun He received his BEng degree in software engineering from Jilin University, China in 2012. Currently, he is an MSc candidate in the Computer Science Department at Jilin University, China. His main research focuses on the detection of botnet.

Xinyang Jiang received his BEng degree in software engineering from Jilin University, China in 2014. Currently, he is an MSc candidate in the Computer Science Department at Jilin University, China. His recent research focuses on the detection of botnet.

Jian Cao received his BEng degree in software engineering from Jilin University, China in 2013. Currently, he is an MSc candidate in the Computer Science Department at Jilin University, China. His recent research focuses on the detection of malicious URLs.

Qiang Li is currently an associate professor in Computer Science at Jilin University, China. He received his BSc, MSc, and PhD degrees also from Jilin University in 1998, 2001, and 2005, respectively. His main research interests are in network security and the detection of malicious code.