HoneySpam 2.0: Profiling Web Spambot Behaviour

Pedram Hayati[*], Kevin Chai, Vidyasagar Potdar, and Alex Talevski

Digital Ecosystem and Business Intelligence Institute, Curtin University,
Perth, Western Australia
{Pedram.Hayati,Kevin.Chai}@postgrad.curtin.edu.au
{v.potdar,a.talevski}@curtin.edu.au

Abstract. Internet bots have been widely used for various beneficial and malicious activities on the web. In this paper we provide new insights into a new kind of bot termed as *web spambot* which is primarily used for spreading spam content on the web. To gain insights into web spambots, we developed a tool (HoneySpam 2.0) to track their behaviour. This paper presents two main contributions, firstly it describes the design of HoneySpam 2.0 and secondly we outline the experimental results that characterise web spambot behaviour. By profiling web spambots, we provide the foundation for identifying such bots and preventing and filtering web spam content.

Keywords: honeyspam, web spambot, web robot detection, spam detection, web usage mining.

1 Introduction

Web content which provides false or unsolicited web information is defined as Web Spam [1]. Spam content is prolific on the web due to the development and widespread adoption of Web 2.0 as spammers no longer need to purchase, host and promote their own domains. Spammers can now use legitimate websites to host spam content such as fake eye-catching profiles in social networking websites, fake promotional reviews, responses to threads in online forums with unsolicited content and manipulated wiki pages etc. This spamming technique which is different from previous Web spamming techniques is referred to as *Web 2.0 Spam* or *Spam 2.0* [2]. Figure 1 illustrates Spam 2.0 and its relation to other spamming campaigns.

Little is known about the amount of Spam 2.0 on the Internet. At the time of writing this paper, *Live Spam Zeitgeist* has shown over a 50% increase in the amount of spam messages (e.g. spam comments in blogging tools) since 2008 [3]. This report showed a dramatic increase in the amount of successful Spam 2.0 attacks and the inadequacy of existing countermeasure tools [2].

In this paper, we conduct an effective study on Spam 2.0 and the behaviour of web spambots. Web spambots are a new kind of Internet robot which is primarily used for spreading spam content on the web. To gain insights into this kind of bot we have developed a tool (HoneySpam 2.0) to track their behaviour.

[*] Pedram Hayati is PhD student at Curtin University.

© Springer-Verlag Berlin Heidelberg 2009

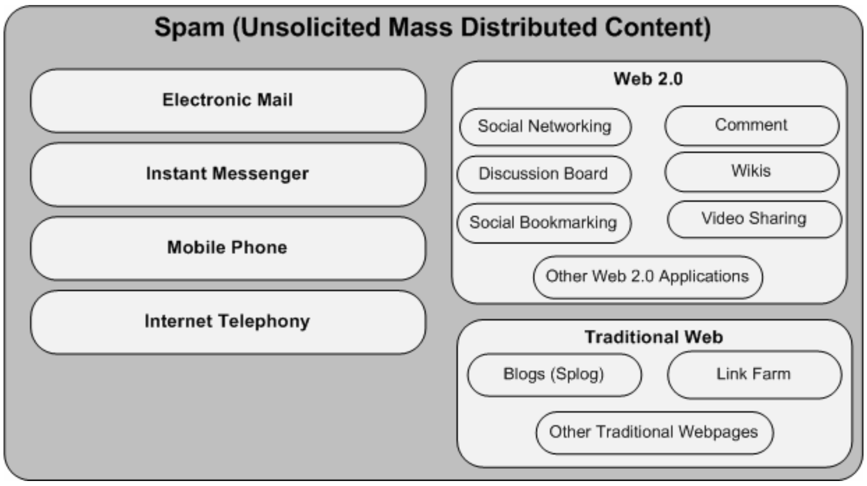


Fig. 1. Spam 2.0 among other spamming campaigns

HoneySpam 2.0 draws on the idea of honeypots (a technique for tracking cyber attackers). It implicitly tracks web usage data which includes detailed monitoring of click streams, pages navigation, forms, mouse movements, keyboard actions and page scrolling. Our results illustrate that HoneySpam 2.0 is effective in profiling web spambots. HoneySpam 2.0 is a tracking solution that can be integrated in any web application ranging from blogging tools to social networking applications. To the best of our knowledge, there is no existing literature that studies web spambots from a web usage perspective. We believe that this work is the first step in finding the attributes of web spambots and is a roadmap for future research.

This paper is organised as follows; in Section 2, we provide taxonomy on Spam 2.0 and web spambots, Section 3 explains the HoneySpam 2.0 architecture. Next, Section 4 discusses our experimental results. Section 5 explains previous work. This is followed by conclusions and future works in Section 6.

2 Taxonomy of Spam 2.0 and Web Spambots

Along with its web applications, Web 2.0 provides a collaboration platform for flexible web content management. Spam 2.0 has existed since web 2.0 concepts have appeared online. Spammers misuse/use this functionality to distribute spam content. The act of spamming can be done through two channels: manual and automated. The former refers to using human operations such as hiring cheap labour to manually distribute spam content [4] (e.g. A sample job advertisement for spam content publishing can be found in [5]). The latter is achieved by exploiting software or scripts which can be in the form of client-side software or web robots. Web or Internet robots are programming scripts or automated agents that are designed for specific tasks such as crawling webpages and parsing URLs without human interaction [6].

Unfortunately, web robots can be exploited for malicious activities such as checking web servers for vulnerabilities, harvesting email addresses from webpages as well as performing Denial-of-Service attacks (DoS) [7]. Recently, some web robots have been used for distributing spam content on web applications through posting promotional comments, placing ads in online forums, submitting links through trackbacks etc. [2]. We call this type of web robots as *web spambots*. Web spambots can be designed to be *application specific* which target specific web applications like Wordpress blogging tools, phpBB, Mediawiki or *website specific* infecting websites like Amazon, MySpace, Facebook etc.

It should be noted that web spambots are different from spambots. According to [7] spambots are Internet robots that crawl webpages to harvest email addresses in order to send spam emails. However web spambots act completely differently from spambots. Web spambots can crawl the web or discover new victims via search engines, register new user accounts and submit spam content in Web 2.0 applications. Their main task is to distribute spam content on to the web. They can reside on different machines (e.g. infected user computers, free or commercial web hosting sites). Web spambots can be programmed once and used many times. The motivations behind such spamming activities are [2]:

- Deceiving search engines to rank spam and junk content higher. For instance, the more spam content added to different websites linking back to the spammer's websites, will increase their website's ranking.
- Misleading users to view unsolicited contents such as webpages, link-farms, ad-hosted website, phishing websites, etc.

By employing automated web spambots, spammers are able to get more traffic on their campaigns. This highlights the importance of web spambot detection as a possible way to eliminate spam 2.0. In the next section we discuss current techniques to discover web spambots along with our proposed solution.

2.1 Web Spambot Countermeasure Techniques

One of the most popular countermeasures to differentiate web robots (include web spambots) from humans is *Completely Automated Public Turing test to tell Computers and Human Apart* (CAPTCHA) [8]. It is a type of challenge-response test usually in the form of a distorted image of numbers and letters, which humans have to infer and type in to a web form. However, CAPTCHA is inconvenient for users as it wastes their time, causes distraction, is unpleasant and at times scary. A number of recent works have reported approaches to defeat CAPTCHAs automatically by using computer programs [9-11]. CAPTCHA's drawbacks includes the following:

1. Decrease user convenience and increase complexity of human computer interaction.
2. As programs become better at deciphering CAPTCHA, the image may become difficult for humans to decipher.
3. As computers get more powerful, they will be able to decipher CAPTCHA better than humans.

Therefore, CAPTCHA is only a short term solution that is about to expire. Web spambots armed with anti-CAPTCHA tools are able to bypass this restriction and spread spam content easily while the futile user inconvenience remains.

Other countermeasures for combating automated request are by using *Hashcash*, *Nonce* and *Form variation* [12, 13]. The idea behind *Hashcash* is that the sender has to calculate a stamp for the submitted content [13]. The calculation of the stamp is difficult and time-consuming for sender but comparatively cheap and fast for receiver to verify. Although *Hashcash* cannot stop automated spamming, it can slow down the spamming process. On the other hand, using *nonce* and *Form variation* techniques can only guarantee that users use their submission forms rather than posting directly.

However, we believe that one effective way to detect web spambot is by looking at web spambot behaviour through its page navigations, click-streams, mouse interactions, keyboard actions and form filling. We put forward the notion that their web usage behaviour is intrinsically different from humans. The focus of this work is on profiling web spambot behaviour. We proposed HoneySpam 2.0 to detect, track and monitor web spambots. In following section we explain our proposed method.

3 HoneySpam 2.0 Architecture

HoneySpam 2.0 is a stand-alone tool that is designed to be integrated with any web application with the aim to track and analyse web spambot behaviour. HoneySpam 2.0 is based upon the idea of a honeypot where a vulnerable server is deployed to attract cyber attackers in order to study their behaviour. HoneySpam 2.0 is designed to study the web spambots which are considered to be attackers in the spamming scenario.

HoneySpam 2.0 consists of two main components: Navigation Tracking Component (NTC) & Form Tracking Component (FTC). Figure 2 illustrates the detailed Model-View-Control (MVC) architecture of HoneySpam 2.0 inside a web application.

3.1 Navigation Tracking Component

This component is designed to track page navigation patterns and header information of incoming traffic. The following information is captured by this component: the time of request, the IP address, the browser identity and the referrer URL.

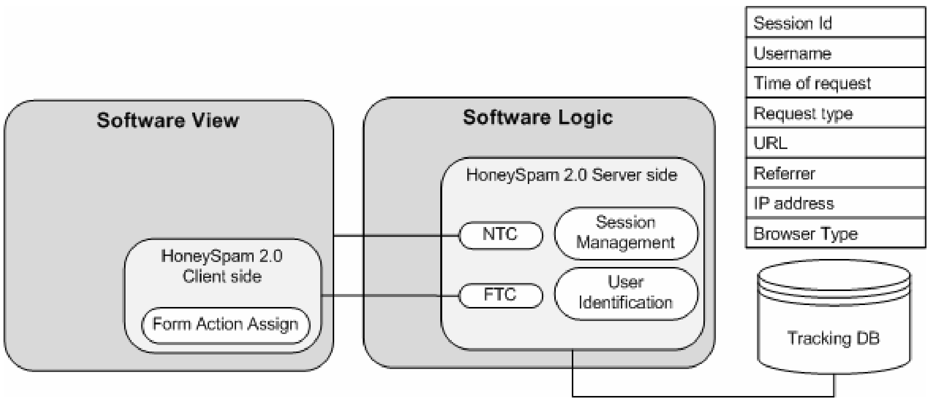


Fig. 2. HoneySpam 2.0 Architecture

NTC is also designed to automatically differentiate individual visit sessions. Session information is captured to identify user's web usage behaviour during each visit. NTC stores the captured data along with session identity (session id) into the tracking database, which is used during an analysis and profiling stage.

3.2 Form Tracking Component

Other than the NTC, we felt the need to develop additional functionality that could provide us insights into how web spambots use web forms. It is a common belief that web spambots do not interact with web forms, however, our study aims to test this here. We developed a form tracking component to specifically study form input behaviour. This component captures and stores the following form related actions such as:

- ⇒ Mouse clicks and movements
- ⇒ Keyboard actions
- ⇒ Form field focus and un-focus
- ⇒ Form load and submission
- ⇒ Page navigation

For each of the above mentioned actions we also captured header information e.g. IP address, browser identity, referrer link session id etc. We decided to store header information since we wanted to ensure that form action events are originating from the same session with the same header information.

Additionally, both components check whether requests are initiated from registered users or visitors. If it was initiated from a registered user then it stores the username along with other web usage information for that request. This was done to differentiate registered web users usage data and visitor usage data, which can be used for further analysis. Visitors are termed as guests in our database. Also, most web applications require the registration of a valid username before further interaction with the system.

3.3 Deploying HoneySpam 2.0

Deploying HoneySpam 2.0 involves implementing both client side and server side code. The client side code, assigns FTC actions to web forms so once a user triggers an action it would be captured, transmitted to FTC server side code and would be recorded in the database. On the server side, NTC tracks the users navigation behaviour based on incoming traffic requests. In the next section, we explain our experimental setting by outlining where we integrated HoneySpam 2.0 and discuss our results and key observations.

4 Experimental Results

The HoneySpam 2.0 tool was employed onto 1 commercial and 5 free web hosting servers. The web applications provided on these servers include online discussion forums and social bookmarking. We did not add any initial content to any of these applications and placed a message (in the form of an image) to notify human users not to use these applications. Additionally, as described in Section 4.4 our FTC component did not capture any form activity records which ensures that there were no human activities inside our hosted campaigns.All of the content added to our web applications is added by web spambots. The data presented in this paper has been collected for a period of 60 days[1] and is summarised in Table 1.

Table 1. Summary of Data for Honeyspam 2.0 showing the total number of bots, total number of posts, average posts per day etc

Summary of Data for Honey Spam	#
No. of tracking records	11481
No. of web spambots' tracking records	6794
Total No. of registered bots	643
Total No. of unique used IP addresses	314
Total No. of posts	617
Average Posts per day	8.27
Average Registered web spambots per day	7.91
Average online web spambot per day	2.72

In terms of content, 60% of spam content was adult-related, 13% consisted of an assorted category of content (e.g. free software, keyword spam) and 9% of content was related to drug (pharmaceutical) and movies (Fig 3a). Demographically, 51% of observed web spambots originated from the United Kingdom followed by the United States of America at 22% (Fig 3b). However, it is quite possible for web spambots to be used by spammers on hijacked machines so these figures do not reveal the true origin of the spammers.

Web browsers commonly used by the web spambots were Internet Explorer and Opera (Fig 3c). Surprisingly, two relatively unknown browsers which were OffByOne and BrowseX were also used. Additionally, Firefox was only used by 2 web spambots and Safari was not used at all. It should be mentioned that this header information can be modified and is not a true indication of browser usage or the use of an actual browser at all. A number of specific observations were discovered from examining the behaviour of web spambots and are now discussed.

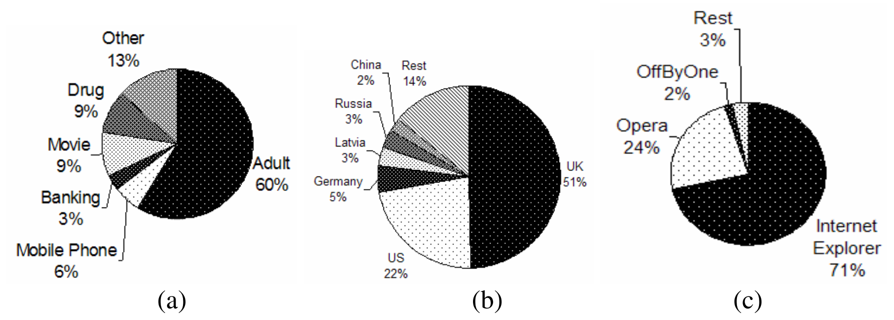


Fig. 3. Percentage of top content categories, countries and internet browsers

[1] The HoneySpam 2.0 tool has tracked spambot behaviour from the 25th of May 2009 to the 9th of August 2009. However, only 60 days of data is collected due to server downtime periods caused by maintenance and issues.

4.1 Use of Search Engines to Find Target Websites

Of the 6 monitored web hosts, the commercial server received the most attention from web spambots. This highlights the possibility that web spambots are using search engine intelligence (e.g. Google PageRank scores) to identify and focus on higher ranked websites that potentially yield more visitor traffic.

The initial 15 days of hosting and tracking did not yield much activity but a number of web spambots began registering and posting content after this period (Fig. 4a). The reason behind this could be related to search engine indexing, which also suggests that some web spambots find their targets through search engine results.

4.2 Create Numerous User Accounts

It was identified from our experiment that web spambots would create a number of new accounts to post their content rather than using their existing accounts. This is shown in Table 1, where the total number of web spambots accounts is 643 but only 314 unique IP addresses are used between these accounts. On average 7.91 new accounts were created per day within our dataset.

It is possible that web spambots adopt this behaviour for additional robustness in delaying account banning (e.g. one user account is associated with numerous spam content items is easier to detect and manage) and/or to handle the event in which their previous accounts are banned.

4.3 Low Website Webpage Hits and Revisit Rates

The observed web spambots adopt similar surfing patterns to one another and commonly visit a number of specific web pages. As illustrated in Fig. 4b most web spambots perform less than 20 page hits in total during their account lifetime.

Fig. 5b supports the claim that web spambots do not revisit many times as 554 web spambots only revisit the websites once and 75 bots revisit less than 5 times. It is likely that these bots revert to creating new user accounts to continue their spamming activities as discussed in section 4.2.

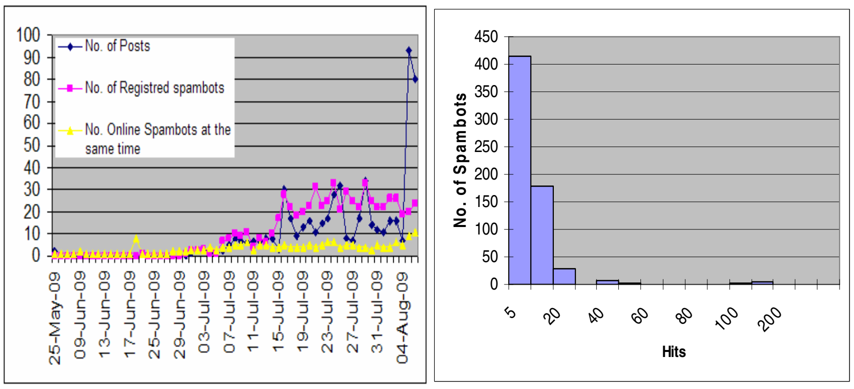
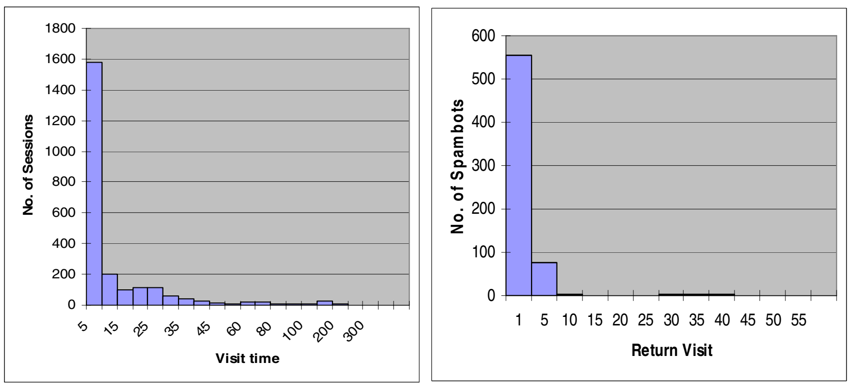


Fig. 4. Number of posts, registered, and online web spambots, frequency of Spambots visits on each Webpages



(a) (b)

Fig. 5. Frequency of Web Spambots' spending time (sec) on each session, frequency of revisiting by Web Spambots (1 means they just visit once)

4.4 Distribute Spam Content in a Short Period of Time

The amount of time most web spambots spend on each visit (session) is less than 5 seconds. This indicates that web spambots are distributing spam content in a short amount of time possibly to move on to other websites to distribute more spam content (Fig. 5a). Additionally their online activitiy time was an average of 2.71 second.

4.5 No Web Form Interaction

While not shown in the figures, the form-tracking component of the HoneySpam 2.0 tool was unable to track form usage behaviour of web spambots. These bots adopt a conventional method of directly posting through their own forms / interface to submit content. We suspect they have not been designed to be intelligent enough to mimic the behaviour of a human in submitting a web form because of a lack of a requirement to do so. Therefore, we believe that simple form usage heuristics can lead to web spambot detection and filtering.

4.6 Generated Usernames

It is apparent that web spambots are generating usernames in order to create numerous accounts. Some examples of these usernames include *Oxidylozaxygixoc*, *Ufigowigelyniwyl* and *Ynutaznazabalekil*. Pattern analysis of these randomly generated usernames could serve as a means of identifying and blocking web spambots, which we will investigate in the future.

5 Related Work

The study of web robots has been thoroughly investigated by the research community since bots consume network bandwidth and make web usage mining difficult. Conventional methods to detect web robots are based on identifying IP addresses and user agents [14]. However, unknown and camouflaged web robots can not be detected by these techniques.

In the web usage mining, Tan and Kumar [6] proposed a model to discover web robots. They use navigational pattern on click-stream data approach to detect web robots. They show that the features such as the request method, the length of the session, the depth and width of webpage coverage of web robots are different from humans. They did not explicitly focus on web spambots as the aim of their work was to detect web robot such as search engine crawlers, offline browsers, link checkers and email collectors. In our proposed work, we concentrate on profiling web spambots which are different to other types of web robots not covered by Tan and Kumar.

By looking at the existence of HTTP requests for CSS and JavaScript files along with mouse movement activities on the Web pages, Park et al. (2006) proposed a technique to detect malicious web robots [7]. The focus of their work was on camouflaged web robots that are normally used for security attacks, harvesting email addresses etc. Similar to previous work, their focus was not on identifying web spambots.

In the area of spam detection, Webb et al. (2008) proposed a social honeypot for detection of spammers in social networking websites [15]. They hosted 51 social honeypots in one of the famous social networking websites. The result of their work shows that temporal and geographic patterns of social spam have unique characteristics.

Andreolini et al. (2005) proposed a method to slow down email harvesting process and poison spamming email databases [16]. Additionally they proposed the creation of fake open proxies to increase the probability of email spam detection.

6 Conclusion and Future Work

In this paper we detailed the design and implementation of HoneySpam 2.0 for detecting, monitoring and analysing web spambots. HoneySpam 2.0 is based upon the idea of honeypots to monitor and characterize web spambots which are a type of web robot that are programmed to surf through the web and distribute the spam content. We integrated Honeyspam 2.0 in popular open source web applications for period of 60 days to monitor web spambot behaviour.

Through our experiments, we discovered that web spambots use search engines to find target websites, create numerous user accounts, distribute spam content in a short amount of time, do not revisit the website, do not interact with forms on the website, and register with randomly generated usernames. By profiling web spambots, we provide the foundation for identifying such bots and filter and removing their spam content.

While this paper focussed on analysing web spambots as a way to combat spam in the future we will focus on adopting clustering algorithms to detect web spambots on the fly. Additionally, artificial neural networks and analytical techniques can be used to gain a better understanding of the data collected from web spambots.

References

[1] Gyongyi, Z., Garcia-Molina, H.: Web spam taxonomy. In: Proceedings of the 1st International Workshop on Adversarial Information Retrieval on the Web, Chiba, Japan (2005)

[2] Hayati, P., Potdar, V.: Toward Spam 2.0: An Evaluation of Web 2.0 Anti-Spam Methods. In: 7th IEEE International Conference on Industrial Informatics Cardiff, Wales (2009)

[3] Zeitgeist, L.S.: Comment Spam. In: Akismet, ed. (2009), http://akismet.com/stats/
[4] Cobb, S.: The Economics of Spam. EPrivacyGroup (2003), http://www.eprivacygroup.com
[5] Workathome, Work from home online ad placing work pay per posting (2009), http://www.workathomeforum.in/online-adplacing-homejob.htm, http://www.workathomeforum.in/online-adplacing-homejob.htm
[6] Tan, P.-N., Kumar, V.: Discovery of Web Robot Sessions Based on their Navigational Patterns. Data Mining and Knowledge Discovery 6, 9–35 (2002)
[7] Park, K., Pai, V.S., Lee, K.-W., Calo, S.: Securing Web Service by Automatic Robot Detection. In: USENIX 2006 Annual Technical Conference Refereed Paper (2006)
[8] Chellapilla, K., Simard, P.: Using Machine Learning to Break Visual Human Interaction Proofs (HIPs). In: NIPS (2004)
[9] Abram, H., Michael, W.G., Richard, C.H.: Reverse Engineering CAPTCHAs. In: Proceedings of the 2008 15th Working Conference on Reverse Engineering. IEEE Computer Society, Los Alamitos (2008)
[10] Mori, G., Malik, J.: Recognizing objects in adversarial clutter: breaking a visual CAPTCHA. In: Proceedings. 2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 1, p I-134-I-141 (2003)
[11] Baird, H.S., Bentley, J.L.: Implicit CAPTCHAs. In: Proceedings SPIE/IS&T Conference on Document Recognition and Retrieval XII (DR&R2005), San Jose, CA (2005)
[12] Ogbuji, U.: Real Web 2.0: Battling Web spam (2008), http://www.ibm.com/developerworks/web/library/wa-realweb10/
[13] Mertz, D.: Charming Python: Beat spam using hashcash (2004), http://www.ibm.com/developerworks/linux/library/l-hashcash.html
[14] Cooley, R., Mobasher, B., Srivastava, J.: Web mining: information and pattern discovery on the World Wide Web. In: Proceedings of Ninth IEEE International Conference on Tools with Artificial Intelligence 1997, pp. 558–567 (1997)
[15] Webb, S., Caverlee, J., Pu, C.: Social Honeypots: Making Friends with a Spammer Near You. In: Proceedings of the Fifth Conference on Email and Anti-Spam (CEAS 2008), Mountain View, CA (2008)
[16] Andreolini, M., Bulgarelli, A., Colajanni, M., Mazzoni, F.: HoneySpam: honeypots fighting spam at the source. In: Proceedings of the Steps to Reducing Unwanted Traffic on the Internet on Steps to Reducing Unwanted Traffic on the Internet Workshop (2005), Cambridge, MA, p. 11 (2005)