The Nuts and Bolts of a Forum Spam Automator

Youngsang Shin, Minaxi Gupta, Steven Myers
School of Informatics and Computing
Indiana University, Bloomington
{shiny,minaxi}@cs.indiana.edu, samyers@indiana.edu

Abstract

Web boards, blogs, wikis, and guestbooks are forums frequented and contributed to by many Web users. Unfortunately, the utility of these forums is being diminished due to spamming, where miscreants post messages and links not intended to contribute to forums, but to advertise their websites. Many such links are malicious. In this paper we investigate and compare automated tools used to spam forums. We analyze the functionality of the most popular forum spam automator, XRumer, in details and find that it can intelligently get around many practices used by forums to distinguish humans from bots, all while keeping the spammer hidden. Insights gained from our study suggest specific measures that can be used to block spamming by this automator.

1 Introduction

With millions of websites on the Internet, attracting visitors to a given site is non-trivial. Website operators, particularly of unsavory sites, are always on the lookout for new mechanisms to make their websites visible. While link embedded email spam continues to be a popular technique for driving traffic to such sites, increasingly, search engines are being exploited as well. The latter activity is so pervasive it is termed web spamming. Specifically, web spamming exploits algorithms used by popular search engines in order to gain better rankings with respect to other sites on the Web. While many tricks are used to achieve the goal, including building link structures favored by search engines, the focus of this paper is *forum spamming*, where miscreants post links to their websites on forums frequented by Internet users. A forum is a website where visitors can contribute content. Examples of forums include web boards, blogs, wikis, and guestbooks. Forums help miscreants in two ways: They help drive traffic to a site directly, and simultaneously increase search-engine rankings for the linked websites. Forums are an attractive target for miscreants because forums with useful content cannot be blacklisted or taken down. While search engines often have a global view of the link structure which permits them to discount some forum spam [23, 14, 11, 9], it does not effectively help forum operators keep their forums free of spam. Ultimately, it is up to forum administrators to remove spam links or prevent their initial posting!

The effects of forum spamming have been studied before [18, 21]. In contrast, we focus on how spammers actually post spam links on forums since understanding their modus operandi can offer important insights for mitigating forum spamming. Toward our goal, we survey forum spam automator tools including XRumer, SEnuke, ScrapeBox, AutoPligg, and Ultimate WordPress Comment Submitter (UWCS) for their functionality. These tools enable forum spammers to automatically post spam to a large number of target forums.

We explore XRumer [7] in detail. It is one of the most popular forum spamming automators on blackhat SEO forums, in fact perhaps the most, and has the most extensive set of features. These include the ability to automatically register fake accounts at target forums, build appropriate browsing history for each forum, and post spam. XRumer is capable of posting at forums built on many software platforms, such as phpBB and vBulletin. Additionally, it can be made to learn new forum platforms it does not directly support. XRumer can also circumvent spam prevention mechanisms commonly employed by forum operators (e.g., automatically solving CAPTCHAs). To keep spammers hidden and to avoid blacklisting, XRumer allows the use of anonymizing proxies which can hide the IP addresses of spamming machines. Further, it can adjust spamming patterns along the dimensions of time and message content to thwart detection. The result is a sophisticated tool designed to bypass most prevention techniques, and one that can adapt to changes in forum software platforms. The picture is not entirely gloomy however, for we find several quirks in XRumer's functionality that we believe can be operationalized into tools that can help mitigate forum spam.

2 Methodology

XRumer has separate demonstration and production versions. The demo version contains the documentation of the full version but has limited functionality. We considered using the production version for this paper as Motoyama et al. in [17] did, testing the economics of CAPTCHA solving. However, we were faced with the moral dilemma that in order to test the efficacy of various production features of XRumer, we would have create fake accounts and post spam on real-world forums, against their terms of use. Short of posting on actual forums, we would simply be posting spam on custom in-house forums, which would preclude testing of various production features in their entirety. This dilemma was not present in Motoyama's work. Therefore, we decided to work with the demo version of XRumer 5.05. It allows registering an account at a forum and then posting a pre-defined message. We could test this on in-house forums we setup for the purpose, avoiding any dilemmas.

We set up our own forum server and observed the packets XRumer generated while posting spam to it. As we describe later in Section 5, this exercise offered useful insights into defeating XRumer. Additionally, we utilized the documentation to gain insights into its other functionalities. For example, one file defines the rules to solve text-based question-and-answer CAPTCHAs. While spammers can add more rules to it, this file permitted us to determine the class of text CAPTCHAs XRumer can solve off the shelf. Another file included information defining how XRumer identifies different forum software. Finally, other files contained various `User-Agent` strings that XRumer can use to impersonate browsers, fake names, addresses, interests etc. This information is used to create and register fake user accounts on varying forums.

The XRumer demo version does not provide for the use of an anonymizing proxy that hides the poster's IP address. To overcome this limitation, we wrote our own code that sends an HTTP request to our forum server by using a public anonymous proxy. This allowed us to examine the change in the HTTP request forwarded by the proxy.

In order to compare XRumer's functionality with other automators, we surveyed the list of functions different automators support based on their websites and tutorial videos available online. This included numerous black-hat search engine optimization (SEO) forums.

3 Primal Functionalities

We begin by describing XRumer's basic functionality, including how it finds target forums, composes spam, and posts it.

3.1 Preparation for Posting Spam

Collecting Target Forums In order to post spam, XRumer needs target forums. A spammer can either provide URLs of target forums or can use `Hrefer`, a free supplement to buyers of XRumer. Hrefer uses search engines to find forums based on a specific list of keywords provided to it as input. A spammer can directly provide the entire keyword list, or provide some initial keywords which Hrefer can use to infer related keywords through the malevolent use of Google AdWords *Keyword Tool* [3]. The keywords play a crucial role in finding thematically coherent forums. This is important since posting a spam message to relevant forums reduces the probability that it will be filtered out.

Hrefer uses the keywords to send search queries. To send search queries, XRumer allows a spammer to choose between Google Web Search, Google Blog Search, MSN, Yahoo, AltaVista, Yandex [8], and board-reader [1]. Each of these search engines provides APIs to automate searches but Hrefer does not use them because APIs provide restricted search results and presumable because they can be used to trace a spammer's activity. So, Hrefer tries to mimic web browser behavior. It parses the returned search results to find target forums.

Composing Spam Message A prudent composition of the spam message is as important for spammers' success as is finding relevant target forums. Thus, all messages must have an appropriate subject. To help defeat message-based filtering, XRumer supports various macros that help create variants of spam messages that are semantically the same but syntactically different. For example, with a simple variation macro, $\{variant\_1|variant\_2|...|variant\_N\}$, chooses one of the N possible variants in the spammed message. The result is that a spammer can create different greetings such as {Hi!|Hello!|What's up!} in the spam message. XRumer leaves the task of choosing appropriate synonyms up to a spam campaign operator. However, the spam automator SEnuke, supports an automated thesaurus of synonyms a spammer can choose from.

In addition to the simple variation macro, a spammer may also use a conditional variation macro based on the forum's environment. For example, the macro, $\{[TLD\_1]\ text\ for\ TLD\_1|...|[TLD\_N]\ text\ for\ TLD\_N\}$, decides the output text based on the top level domain (TLD) of the forum used to post spam. This permits a large amount of contextualization. For example, {`#.com` Hi!|`#.fr` Bonjour!}, results in an English or French salutation depending on the `.com` or `.fr` TLD. Table 1 provides the full list of macros supported by XRumer. Multiple studies, including [15, 22, 19], have identified the use of macros in the context of email spam. In particular, Pitsillidis et al. in [19] explore how to model variant spam messages generated from the same spam botnet by inferring their templates.

XRumer recommends spammers to use Bulletin Board Code (BBCode) in composing spam messages, particularly for embedded links. BBCode is a simple markup language used to format posts in many forums. For example, using `[URL]`...`[/URL]` transforms the inserted URL into the corresponding anchor tag, `<a href="...">...</a>`, in the forum's HTML rendering. This enables forum visitors to easily follow a link by clicking on it. However, the support of BBCode depends on the target forum software and its configuration. Should the forum not support BBCode, then spammer's message would simply be rendered as text. XRumer does not validate that the forum supports BBCode.

Macro	**Function**
{variation_*1*|variation_*2*|...|variation_*N*}	One variation is chosen as an output
{**#[identifier]** variation*1*_*1*|variation*1*_*2*|...|variation*1*_*N*}··· {**#[identifier]** variation*M*_*1*|variation*M*_*2*|...|variation*M*_*N*}	Co-variation by [identifier]
{**#[TLD*1*]** variation_*1*|**#[TLD2]** variation_*2*|variation_*3*|...|variation_*N*}	Variation by [TLD]
[color=color_url]...**[/color]**	Sets the style of message text to be the same as the link, so any visual difference between links and text is eliminated
#category, **#hostname**	Replaces word with a category name or host name respectively
#random[range of variants]	Randomly chooses one of the variants within the specified range
#file=filename	Replaces word with the content of the specified text file
#gennick[identifier] or **#gennick**[identifier, min_length, max_length]	Generates a nickname with spammer's controlled lengths following the domain name and identifier
#file_links[filename, num_of_lines, formation_method]	Replaces the specified number of lines from the specified file
#err[error_maker]	Generates typos, with a higher error_maker generating more typos
#nomacros...**#endnomacros**	Disables all enclosed macros

Table 1: Macros supported by XRumer

3.2 Posting Spam

XRumer has a built-in database that allows it to post to various types of forum software: `phpBB`, `PHP-Nuke`, `yaBB`, `vBulletin`, `Invision Power Board`, `IconBoard`, `UltimateBB`, `exBB`, `phorum.org`, `livejournal.com`, `AkoBook`, and `Simple Machines Forum`. Importantly, it provides tools to aid in the automated posting of spam to new or proprietary forum software (cf., Section 3.3).

Registration In order to deter automatic postings most forums require their visitors to register before they can post a new message or comment. The registration process often involves account activation over email and/or solving a CAPTCHA. XRumer is built to overcome these barriers. If a spammer provides email accounts, XRumer can use them to register fake forum accounts. It mechanically visits targeted forums, fills out necessary forms, and completes the activation process by processing any standardized activation mail received in the email accounts. If the forum requires CAPTCHAs to be solved during the registration process, XRumer tries to solve them automatically using built-in algorithms, or permits their solution to be seconded to professional CAPTCHA solving services (cf., Section 4.1). If no email addresses are provided, XRumer will automatically create email accounts on GMail for automated registration.

Posting To post spam, XRumer logs into the site as a registered user. On forums with multiple topics or discussions, XRumer uses a *priority categorization* to determine which topic or discussion to post to. The priority category is nothing but a rating from one to three. XRumer's first priority is to post spam under forum topics enumerated by the spammer. For example, to post spam on subjects related to "Real estate", the spammer might give keywords such as "real estate", "to lease", "to rent", "rent", "lodging", or "apartment". XRumer looks for forum topics containing these keywords and attempts to post to them. If it cannot find any such topic in the target forum, it tries to find default topics in the second priority category. XRumer provides a (spammer modifiable) generic list of forum topics, such as "Off topic", "Flame", "Flood" and "Advertising". If XRumer fails to find any of the generic categories, its last priority is to post spam at the most visited forum topic. The last priority posting actually has the highest importance for specific types of forums, such as blogs. This is because blogs usually do not have topic or category postings. Our recent study on the prevalence and mitigation of forum spamming [21] confirms this strategy, as we showed that popular posts receive more spam than unpopular ones.

Just as in the account registration phase, XRumer can solve CAPTCHAs during the spam posting phase (cf., Section 4.1). Further, most forums now provide a function allowing their users to create a poll for other users to rate their contributions. XRumer can activate such polls for their postings. We believe this may help spam postings get attention from other forum users, and lead to an increased belief in the legitimacy of the spammer's user account.

Refspam Forums are almost universally implemented as server side scripts using scripting languages, such as PHP, ASP, JSP, or CGI. This allows recording of traffic logs just like regular web traffic. A typical log entry includes access time, client IP address, page accessed, and browser version. Some applications, such as *Webalizer* [5], collect and manage more information about traffic to the web server. This often includes the `Referer` HTML header, which contains information about the URL a client visited prior to visiting the current URL.

When XRumer visits a forum on a web server configured with *Webalizer*, it can insert a `Referer` header with the spammer's target malicious URL in the HTTP request's `Referer` field. The beneficial result for the spammer is that even if the spammer's post is removed, *Webalizer* would record the spammer's `Referer` link and publish it in a reporting page on the forum server. When a search engine bot visits the reporting page, it considers the `Referer` URLs to be outgoing links from the forum server. As a result, the spam URL will be viewed as an outgoing link from the forum server, gaining authenticity. Thus, if the forum has a high search engine rank, the spammer's website will benefit from an increase in its own page rank. This is referred to as *refspam* in XRumer. Fortunately, search engines can counter this technique by segregating such reporting pages during their crawls, and forum operators can assist in such processes by putting `ref`="`nofollow`" into the page's html file, or by putting "`disallow`" for such pages in `robots.txt`.

3.3 Advanced Spam Posting

Self-Learning Many forums are built on proprietary software whose input schema are unknown to XRumer. For such cases, XRumer provides a *self-learning* function. The title slightly overstates its functionality. If a spammer activates this function, XRumer collects unknown HTML form inputs while trying to post spam on a given page and returns them. Specifically, it collects inputs with their name, data-type, label-text that appears next to the input on the form, and the form's source URL. The spammer can then specify the responses that XRumer should give to these input forms. Although XRumer does not automatically determine the semantic meaning of unknown inputs (as self-learning might suggest), spammers can use the retrieved information to facilitate the process of determining an appropriate response expected by the given form.

Reporting XRumer's rate of successfully posting spam depends on the spammer's inputs, including the spam message and keywords for finding appropriate forums. To judge its success, XRumer analyzes the HTML page returned upon its request to post. It generates a report which shows overall success rates by TLD, forum software, or both. It also analyzes patterns of URLs where spam was successfully posted and allows the user to filter out posting attempts based on various conditions.

4 Detection Avoidance Techniques

XRumer provides various measures to defeat common counter-measures used by forums to identify forum spam. We describe them in this Section.

4.1 Solving CAPTCHAs

XRumer can solve text and graphical CAPTCHAs, the two most common forms of CAPTCHAs in use. For text CAPTCHAs, XRumer has a number of rules and predefined responses to answer common questions. Specifically, it can solve three types of questions. The first type includes arithmetic operations such as "what is the answer for 2+3=?". The next type asks a visitor to type the displayed phrase. The last type has trivia questions, for example "What is the capital of the USA?" To answer such CAPTCHA questions, XRumer has a list of question and answer pairs. The first two types of text based CAPTCHAs are easily resolved by parsing and semantic understanding. The last group is solved by using a lookup table consisting of a list of pairs of common questions and their responses. In all three cases, XRumer's ability to solve text CAPTCHAs can be modified through a configuration file. In the first two cases, rules can be used to program XRumer to solve more complicated problems of arithmetic or retyping. Question and answer pairs can also be added.

The success rate of XRumer in solving graphical CAPTCHA depends on their type. Work by Motoyama et al. investigated this issue and found that XRumer's CAPTCHA solvers targeted "weaker" CAPTCHAs and achieved an accuracy of 100% in some cases, with a response time of under a second [17]. By default, XRumer tries to solve CAPTCHAs until it is successful. However, since solving CAPTCHAs hurts performance, XRumer allows control over the number of failed solving attempts before giving up. While many forums track IP addresses of failed CAPTCHA attempts for blacklisting purposes, the use of anonymizing proxies makes this a non-concern for most spammers. It is worth noting, that recent upgrades in popular forum software have resulted in CAPTCHAs that are impervious to XRumer's solver, with the exception of Simple Machines Forum [17].

For more complicated graphical CAPTCHAs, XRumer provides two alternatives: a spammer intervention mode and a subcontracter mode. In the spammer intervention mode, XRumer presents the irresolvable CAPTCHA to the spammer after a predefined number of failures so the spammer can solve it manually. In the subcontract mode, XRumer allows spammers to use online CAPTCHA solving services such as Anti-Captcha and CaptchaBot. Both of these services are currently offering to solve CAPTCHAs at a cost of $\approx$\$1 U.S. per 1000 CAPTCHAs. In addition to these alternatives, XRumer provides an SDK that a spammer capable of programming can use to write a solving library in the form of a DLL.

4.2 Question and Answer

This function allows a spammer to post to a forum both a question in one post and its answer in another from a different account. This feature has two purposes. The first is to disguise a spam message as a solicited answer, making it hard for a forum moderator to easily block such postings. The second goal of this functionality is to build a good activity history for fake accounts. This helps build history for forums that disallow URLs in messages or signatures until a user has made 5$\sim$10 legitimate posts.

4.3 *Antispam*

XRumer provides another function for building a good history. It is called the *antispam* function. This function randomly chooses postings asking a question and tries to find a thematically relevant answer in other forums on the Web. Posting such answers can help a spammer build a good activity history. This is a relatively new feature of XRumer, still officially in a beta phase for XRumer v5. However, the existence of this function shows the level of sophistication the developers of the automated spamming software are attempting to achieve.

4.4 Anonymizing Proxies

In order to hide the IP address of spamming machines, XRumer allows a spammer to set up an anonymizing proxy for XRumer as well as Hrefer. With privacy being a concern in the Internet today, both free and paid anonymizing proxies exist and are being used by various applications already. XRumer and Hrefer simply simplify their use.

XRumer provides a list of anonymous, free public proxies. However, this list is not necessarily useful since the lifetime of free proxies is typically short. For this reason, XRumer recommends that its users use a list of paid proxies for better performance in terms of speed, uptime and effectiveness of anonymization. To help, XRumer includes pointers to lists of public proxies. Irrespective of the type of proxy used, XRumer verifies each proxy for anonymity and then saves a list of ones that pass the test. For checking anonymity, XRumer uses a PHP script that when installed at a (controlled) web server, shows HTTP headers in the HTTP request sent by an anonymizing proxy. If the proxy exposes the IP of the sending client, XRumer does not use that proxy. XRumer refreshes the proxies to be used regularly, by default every 30 minutes.

4.5 Spam Traffic Control

XRumer provides various options for adjusting traffic by trading off spamming speed and rate of postings. For example, the followings are all configurable parameters in XRumer: the maximum size of forum pages, the maximum number of links in forum pages, GET- or POST-query timeout, and the number of maximal attempts to solve CAPTCHAs. Spammers can tune these parameters to cause distinct and hard to recognize traffic patterns. Further, XRumer supports a scheduler, so when a certain event happens (e.g., *posting finished*, *a timer goes off*, *the number of successful postings reaches a preset limit*), XRumer can schedule the execution of specified actions.

5 Traffic Characteristics

We used XRumer to post spam to forums we set up for experimentation. In this section, we discuss traffic characteristics we observed through this exercise. We also discuss how they can be leveraged to defeat XRumer.

5.1 HTTP header

The first thing we investigated was the HTTP headers generated by a client machine running XRumer, as observed at a forum web server. The configuration of client and server used for this experiment is shown in Table 2. To observe the generated HTTP traffic, we used Wireshark [6].

Program	Role
XRumer 5.05 demo	Forum spam automator running at the client
Internet Explorer (IE) 6	Client web browser
MS Windows XP without any service pack patch	Client OS
phpBB 3.0.7	Forum software running on the web server
Apache HTTP server 2.2	Web server
Linux (Kernel 2.6.25)	Server OS

Table 2: The configuration of client and server for the HTTP request observation.

Figure 1 shows the HTTP headers generated by the client browser. The `GET` or `POST` indicates that this request is a HTTP GET or POST request respectively. Presence of `HTTP/1.1` says that the web client uses HTTP protocol version 1.1, which supports persistent TCP connections, unlike HTTP version 1.0. `Accept`, `Accept-Language`, and `Accept-Encoding` show acceptable content type, language of the response, and its encoding respectively. The `User-Agent` header is used to identify the HTTP client and its operating system (OS). Its value depends on the version of IE and the version and setting of Windows, such as the presence of a service pack. The `Host` header indicates the name of web server. `Connection` tells if the web client wants to keep the TCP connection open or not: a `Keep-Alive` indicates a persistent connection while a `Close` not. The `Cookie` header contains the cookie.

```
GET or POST {path} HTTP/1.1
Accept: */*
Accept-Language: en-us
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0;
Windows NT 5.1)
Host: {forum host name}
Connection: Keep-Alive
Cookie: {cookie}
```

Figure 1: Sample HTTP headers generated by Internet Explorer 6 in MS Windows XP without any service pack. If the web server is not running on port 80, `Host` is *host-name:port number*. The specification of HTTP header except that for `Cookie` can be found in [13]. The details of `Cookie` can be obtained at [16].

XRumer puts six headers into its HTTP request, as shown in Figure 2. It uses `HTTP/1.0` while virtually all modern web clients use `HTTP/1.1`. It is not clear why XRumer still chooses `HTTP/1.0` over `HTTP/1.1` since there is no obvious advantage to using HTTP/1.0. There are two headers that are different because of HTTP/1.0 usage. First, the Host header is not supposed to be present in an HTTP/1.0 request but is present. Furthermore, in our experiment, we use a special port number for our web server instead of the standard port 80. IE sets Host to *hostname*:*port number*, but XRumer sets it just to *hostname*, without the port number. *The existence of Host header and its unconventional usage in the HTTP request indicate that this set of HTTP headers is not from a legitimate web client and can be used to spot present-day XRumer versions.*

```
GET or POST {path} HTTP/1.0
Accept:  */*
User-Agent: {User-Agent string}
Referer: {visiting URL}
Host: {forum host name}
Proxy-Connection:  Keep-Alive
Cookie: {cookie}
```

Figure 2: HTTP headers generated by XRumer

Another noteworthy header due to HTTP/1.0 usage is Proxy-Connection. Although Proxy-Connection is not a standard HTTP header for either HTTP 1.0 or 1.1, it was implemented in some versions of web browsers such as Netscape Navigator [4]. The HTTP 1.0 web clients supporting Proxy-Connection put the header to manage a persistent connection through a web proxy even though it works only if the web proxy supports it. *XRumer seems to assume that a spammer would use an anonymizing proxy to hide the IP address, hence it uses the Proxy-Connection header even though this header is not a standard header.* Again, this feature can be used to detect XRumer.

XRumer's first HTTP requests to new web servers should not have the Cookie header present since no cookie from the server exists. However, *XRumer adds the Cookie header with an empty value even in its first HTTP request.* This feature can also be used to spot present-day XRumer versions.

The Referer header generally contains the URL of the web page from which a user followed a link to arrive at the current page. There is no Referer in Figure 1 header because the site was visited by directly typing the URL in to the web-browser, and thus no link was followed. However, XRumer headers always have a Referer header. Furthermore, the header's content is unusual as compared to normal web browsing, for XRumer sets it to the currently requested URL by default and only to the spammer's advertised link if the *refspam* option is turned on. We conjecture that XRumer does this to make its postings appear legitimate, because for a typical posting at a forum occurs upon following a link and thus should have a Referer header. On the other hand, this would rarely be the current page. In fact, some forum platforms, including phpBB, have an option to check if the URL in Referer is valid in terms of availability, but they cannot validate if the URL in Referer is semantically correct because any web site can have a link to the currently requested URL.

The User-Agent header can be used by the web server to infer if a visitor is a bot or human because a User-Agent belonging to a browser is taken to imply a human being is visiting. Search engine crawlers typically use different User-Agent strings. To get around any such checks done by forum servers, XRumer inserts a User-Agent string belonging to one of the popular web browsers in an attempt to make its postings look like they were made by a human. In fact, it changes the strings for different sessions of postings to look like different web browser each time. Examples of User-Agent strings are shown in Figure 3. A data file, x_user_agent.txt, containing User-Agent strings shows that it can send a string for MS IE 3.02 to 7.0, Mozilla 0.6 to 6.0, or Opera 7.11 to 9.01. While previous features will help detect XRumer, this will make it difficult to identify XRumer.

- Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.2; SV1; .NET CLR 1.1.4322; .NET CLR 2.0.50727)
- Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; FREE; .NET CLR 1.1.4322)
- Mozilla/4.0 (compatible; MSIE 5.5; Windows NT 5.0)

Figure 3: User-Agent string examples by XRumer.

5.2 Proxy Usage

As described in Section 4.4, XRumer allows spammers to use anonymizing proxies while posting at forums. We wrote our own code that used free public proxies to connect to our forum server in order to understand various aspects of proxy usage. In general, there are four different types of proxies available in the Internet:

- **Transparent proxies:** They identify themselves as proxy servers by exposing the original IP address of the poster in HTTP headers.
- **Anonymous proxies:** They hide the original IP address while admitting that they are a proxy server.
- **Distorting proxies:** Such proxies expose themselves as a proxy, but put an incorrect client IP address in place of the original IP address.
- **High anonymity proxies:** Such proxies try to hide client IP address and the fact that they are a proxy.

For XRumer, *high anonymity proxy* would be the best choice since it does not expose the existence of a proxy server in addition to hiding the original IP address. However, except *transparent proxy*, any of the other three types of proxies can be used by XRumer since they all hide the original IP address. We refer to all of those three types of proxies as *anonymous proxies* subsequently.

Though XRumer supports two types of proxies, HTTP and SOCKS, we experimented only with HTTP proxies, as without the professional version of XRumer we could not determine how XRumer actually connects to SOCKS proxies, and therefore could not emulate it. Both types of proxies are often referred to as *web proxies*. We use the terminology to refer to anonymizing proxies used by XRumer in the rest of this paper. Anonymous web proxies are of two types. The first type works only as web services. A user visits their website and puts the URL that they want to visit through the proxy. The web service forwards the request to its own proxy and gets the page from the URL. Furthermore, the resultant pages are modified in that they often contain advertisements or may alter links on the original web page to ensure semantic consistency of links through the proxy. XRumer cannot use web service proxies directly. The second type of proxy is an open port proxy. Such proxies provide their IP addresses and port numbers so that any application can use them. We experimented with the second kind. Specifically, we examined how HTTP headers are changed by the proxies, and if proxies forward traffic directly to the target host or to another internal proxy for load balancing. Since the default proxy list provided by XRumer was not valid, we collected a list of public anonymous proxies from `http://www.xroxy.com`. Of the 165 anonymous proxies listed there, 105 were available at the time of our experiment.

To access web proxies, we wrote our custom web client in Python. Then, we sent an HTTP request to our web server through the proxies. We inserted only the following HTTP headers: `Accept`, `Accept-Language`, `Accept-Encoding`, `User-Agent`, `Host`, `Connection`, and `Referer`, which are the headers sent by MS IE 6 except `Referer` in Figure 1. 55 proxies did not add any additional HTTP header. The remaining 50 proxies added one or more HTTP headers listed in Table 3. One interesting HTTP header is `Accept-Encoding`. The Python `urllib2` package sent its value as "identity", while 45 proxies actually removed the `Accept-Encoding` header. Among 60 proxies sending the header, 9 proxies changed its value from 'identity' to 'text/html, text/plain'. Most modern web browsers send an `Accept-Encoding` header with a `gzip` compression value [20]. This information can be used to detect if an incoming HTTP request is through a proxy in many cases.

We also examined if the anonymous web proxies sent incoming traffic directly to the web server. Among 105 proxies, almost a half (54) did that. The others forwarded incoming traffic to other intermediate proxies.

HTTP header	# of proxies
Cache-Control	49
Keep-Alive	1
X-Bluecoat-Via	3
X-Forwarded-For	1

Table 3: HTTP headers inserted by public anonymous proxies and the number of proxies adding each header

6 Comparison with Other Forum Spam Automators

We compare XRumer with the other forum spamming automators including SEnuke, ScrapeBox, AutoPligg, and UWCS in terms of their functions. [1] The basic spam posting functions that XRumer, SEnuke, ScrapeBox, and AutoPligg support are similar while UWCS's functions are comparatively primitive. The main difference is the forum platforms supported. While XRumer can post spam to forums built on various forum platforms, ScrapeBox supports only three platforms, while AutoPligg and UWCS can spam only one platform, Pligg and WordPress respectively. ScrapeBox and UWCS support only blog platforms. Thus, they do not provide an automatic registration function since many blogs do not require their visitors to register in order to leave a comment. SEnuke targets a number of popular forum services, but focuses more on creating *splogs*. Splogs are spam blogs whose sole purpose are to have spam posted to them. Most of the automators, with the exception of UWCS, present macro support for writing syntactically different spam messages. SEnuke even offers an automatic spam message generation tool for an additional fee. No other automator offers this feature, including XRumer.

The advanced functions are where XRumer's sophistication stands out. While all the automators we surveyed report on the results of their activity and support anonymizing proxies, only XRumer allows various reporting options and can be modified to post on new, unsupported forum platforms. Furthermore, only XRumer has the functionality for building legitimate posting histories and for controlling spam traffic with various options. Finally, all automators except UWCS provide integration with CAPTCHA solving services.

7 Related Work

Spam in general and forum spam in particular has been studied [18, 21]. However, tools used by spammers are less studied. Cova et al. analyze phishing toolkits used to build phishing sites in [12]. Their study throws light on miscreants' modus operandi from a different perspective than ours. Motoyama et al. investigated the economics of CAPTCHA solvers, including this aspect of XRumer [17]. Their study focused largely on paid CAPTCHA solving services.

Detection of software similar to XRumer or another forum spam automators has thus far not been studied. Works in [10, 2] propose methods for detecting proxy usage in general which can be helpful in detecting the use of anonymizing proxies by forum spam automators. These works use skew in server response time, patterns of TCP acknowledgments, and packet inter-arrival times to achieve their goal.

8 Conclusion

We investigated various features of a popular forum spam automator, XRumer, in this paper. We found that XRumer takes many steps to defeat common measures forum operators take to dissuade misuse. It can also keep spammers hidden, making detection even more challenging. Consequently, our study offers important lessons for why current methods to protect forum misuse are inadequate.

We also find that a few features of current XRumer versions can help fingerprint and detect its postings. For example, search engine pages can identify XRumer's use of *refspam*, as we discussed in Section 3.2. Further, XRumer uses HTTP headers in unusual ways, which can aid in detecting XRumer's postings. Its use of anonymizing proxies can also be detected, simply by learning the IP addresses of free and paid anonymizing web proxies available. However, many XRumer users probably have no moral issues in using botnet-based proxy services, in which case the blacklisting of proxy services would be worthless. Further, certain forums that require anonymity (e.g., dissident forums) may very well require legitimate postings from proxy services. While we were able to investigate the key features of XRumer through the demo version, the unavailability of Hrefer limited our study, in that we were unable to observe how it collects forums for spamming.

In this paper, we do not intend to, nor can we, quantify how effective is XRumer's in its act of spamming forums. This is because its effectiveness depends highly on forum spammers' SEO knowledge, which is the basis of successful spam contents and links. Furthermore, while our findings can help mitigate forum spam in the short run, XRumer can adapt to make this cat-and-mouse game more difficult by adapting to the above countermeasures. As it is, automators, including XRumer, evolve aggressively not only to improve their success rates, but also to better avoid the deployed countermeasures. For the long run, a promising approach seems to be to make forums intentionally divert from homogeneous registration and posting forms, making it impossible for automators such as XRumer to build databases of expected forum functionalities.

Acknowledgment

The authors would like to thank Patrick Fitzgerald from Symantec for providing materials on XRumer and for proof-reading the manuscript.

References

[1] Boardreader. http://www.boardreader.com.

[2] Forensics wiki - proxy detection. http://www.forensicswiki.org/wiki/Proxy_server#Proxy_detection.

[3] Google AdWords keyword tool. https://adwords.google.com/select/ KeywordToolExternal.

[4] Mozilla HTTP handler, nsHttpHandler.cpp source code. http://bonsai.mozilla.org/cvsblame.cgi?file=mozilla /network/protocol/ http/src/nsHttpHandler.cpp&rev=1.129#387/.

[5] Webalizer. http://www.mrunix.net/webalizer/.

[6] Wireshark. http://www.wireshark.org.

[7] XRumer. http://www.botmasternet.com.

[8] Yandex. http://www.yandex.com.

[9] ABERNETHY, J., CHAPELLE, O., AND CASTILLO, C. Web spam identification through content and hyperlinks. In *WWW AIRWeb* (2008).

[10] CANINI, M., LI, W., AND MOORE, A. W. Toward the identification of anonymous web proxies. In *PAM* (2009).

[11] CASTILLO, C., DONATO, D., GIONIS, A., MURDOCK, V., AND SILVESTRI, F. Know your neighbors: Web spam detection using the web topology. In *ACM SIGIR* (2007).

[12] COVA, M., KRUEGEL, C., AND VIGNA, G. There is no free phish: An analysis of "free" and live phishing kits. In *USENIX WOOT* (2008).

[13] FIELDING, R., GETTYS, J., MOGUL, J., FRYSTYK, H., MASINTER, L., LEACH, P., AND BERNERS-LEE, T. Hypertext Transfer Protocol – HTTP/1.1. RFC2616, 1999.

[14] GAN, Q., AND SUEL, T. Improving web spam classifiers using link structure. In *WWW AIRWeb* (2007).

[15] KREIBICH, C., KANICH, C., LEVCHENKO, K., ENRIGHT, B., VOELKER, G. M., PAXSON, V., AND SAVAGE, S. On the spam campaign trail. In *USENIX LEET* (2008).

[16] KRISTOL, D., AND MONTULLI, L. HTTP state management mechanism. RFC2965, October 2000.

[17] MOTOYAMA, M., LEVCHENKO, K., KANICH, C., MCCOY, D., VOELKER, G. M., AND SAVAGE, S. Re:CAPTCHAs - understanding CAPTCHA-solving services in an economic context. In *USENIX Security Symposium* (2010).

[18] NIU, Y., WANG, Y.-M., CHEN, H., MA, M., AND HSU, F. A quantitative study of forum spamming using context-based analysis. In *NDSS* (2007).

[19] PITSILLIDIS, A., LEVCHENKO, K., KREIBICH, C., KANICH, C., VOELKER, G. M., PAXSON, V., WEAVER, N., AND SAVAGE, S. Botnet judo: Fighting spam with itself. In *NDSS* (2010).

[20] SCHRÖPL, M. Which browsers can handle content-encoding: gzip? http://schroepl.net/projekte/mod_gzip/browser.htm.

[21] SHIN, Y., GUPTA, M., AND MYERS, S. Prevalence and mitigation of forum spamming. In *IEEE INFOCOM* (2011).

[22] STERN, H. A survey of modern spam tools. In *CEAS* (2008).

[23] ZHOU, D., BURGES, C. J., AND TAO, T. Transductive link spam detection. In *WWW AIRWeb* (2007).

Notes

1. We surveyed XRumer 5, SEnuke 6, ScrapeBox 1.14.6, AutoPligg 5, and UWCS 2.5 at the time we wrote this paper. However, the authors of these tools have been actively updating their functionalities. Thus, some restrictions of each tool might be no longer valid.