TrackBack Spam: Abuse and Prevention

Elie Bursztein*
Stanford University
elie@cs.stanford.edu

Peifung E. Lam*
Stanford University
pflam@cs.stanford.edu

John C. Mitchell*
Stanford University
mitchell@cs.stanford.edu

ABSTRACT

Contemporary blogs receive comments and TrackBacks, which result in cross-references between blogs. We conducted a longitudinal study of TrackBack spam, collecting and analyzing almost 10 million samples from a massive spam campaign over a one-year period. Unlike common delivery of email spam, the spammers did not use bots, but took advantage of an official Chinese site as a relay. Based on our analysis of TrackBack misuse found in the wild, we propose an authenticated TrackBack mechanism that defends against TrackBack spam even if attackers use a very large number of different source addresses and generate unique URLs for each TrackBack blog.

1. INTRODUCTION

As users place more and more of their personal information on cloud computing sites, the need for links and cross-references between such sites also increases. One established example is blog cross-referencing, which is so important that automated methods have been developed to facilitate cross references.

Since their emergence over a decade ago [17], blogs have become a major form of communication, with more than 184 million blogs read by more than 346 million readers in 2008 [16]. Along with widespread legitimate use for communicating information and opinions, blogs have naturally attracted two forms of spam (unwanted postings): *comment spam* and *TrackBack spam*. Comment spam is an extension of traditional email spam and can be mitigated by requiring users to authenticate before commenting, and CAPTCHAs [2]. On the other hand, *TrackBack spam* is specific to blogs.

*This work was partially supported by the National Science Foundation and a MURI grant administered by AFOSR.

CCSW'09, November 13, 2009, Chicago, Illinois, USA.

The TrackBack mechanism [3] is used to automatically insert cross-references between blogs. A new blog post citing an older one on a different blog can use the TrackBack interface to insert a link in the older post *automatically.* TrackBacks are an intrinsic part of the blogosphere, and a key ingredient used in blog ranking (Sec. 2). Because TrackBacks are automated, CAPTCHA and registration requirements cannot be used to protect the TrackBack mechanism. Over the last few years, abuse of the TrackBack mechanism has emerged as a key problem, with some attacks causing sites to disable this feature [5]. So far, however, very little research has been conducted on how TrackBack spam is carried out in the wild. To better understand how attackers currently abuse the TrackBack mechanism and to help design better defenses in the future, we instrumented an operating blog site and collected TrackBack spam over a one-year period. This paper reports the results of our longitudinal study of a blog spam campaign, perhaps the first publicly recorded, providing actual data about attacks and their results. To perform this study we have:

- **Developed a honeyblog**: We created and maintained a real blog with a custom TrackBack mechanism designed to collect spam samples and deceive the spammer.
- **Collected and analyzed millions of samples**: Over one year, we collected and analyzed almost 10 million spam samples. We have analyzed trends such as evolution in the reported user agent, the number of IP addresses used, and the content of the spam itself.
- **Performed a network analysis**: We used OS passive fingerprinting and IP geolocation to investigate the spam platform and its operation.

One surprising conclusion of our study is the nature of the platform used by blog spammers. Based on the trend for email spam [13], we expected spammers to use bots. However, it turned out that instead they used stable long-lived servers located in Russia.

Another interesting result from this study is that spammers were able to take advantage of a Chinese official web site as a relay, until its webmaster decided to improve its security by requiring Chinese CAPTCHAs.

We believe that TrackBack spam, including spam delivered using the specific attack methods we observed in the wild, can be prevented using an authenticated extension of the TrackBack protocol. Since a spammer may try to attract readers to malicious sites by sending a large number of TrackBacks to a large number of blogs, without flooding any single blog, we believe that the only effective defenses will be those that collect and aggregate information from many blogs. We therefore propose a defensive architecture in which participating blogs register with a central authority, perhaps implemented by distributed means.

In the most straightforward approach, the authority certifies public verification keys associated with cryptographic signing keys generated by each blog site. This public-key infrastructure allows TrackBacks to be signed by their senders and verified by receiving blogs or the anti-spam authority. While the main function of the authority is then to count the number of TrackBacks generated by each signing principal, there are a number of additional issues that can be addressed to make this mechanism more effective, resistant to misuse, and flexible enough in granting different TrackBack privileges to different users to avoid interfering with honest blog activity. For example, senders may request TrackBack budgets through several alternative means, regulated using CAPTHCAs or other methods, and a receiving blog may determine that TrackBacks are spam or not according to their own policy. While signatures provide the most direct and reliable approach, it is also possible to achieve similar effects through shared symmetric-key infrastructures.

The remainder of the paper is organized as follow: In sec 2, we discuss the blogosphere and blog spam. In sec. 3, we describe our honeyblog setup and statistics. In sec. 4, we analyze data about the spamming platform. In sec. 5, we examine the content of the spam captured by our honeyblog. In sec. 6, we describe the lure site and its operation. In sec. 7, we outline our plan to develop Talkback, a new trackback mechanism designed to mitigate trackback spam. In sec. 8, we present alternative approaches to Talkback and discuss their limitations. In sec. 9, we summarize additional related work. Finally, we conclude in sec. 10.

2. BACKGROUND

In this section we summarize relevant aspects of the Blogosphere, how the TrackBack mechanism works, how it is abused by spammers, and some reasons why they do so.

Blogosphere.

The term *Blog* is a contraction of the term *web-log*. Blogs can be used for any topic, but are often used by *bloggers* to share and exchange information and personal opinions on subjects that range from personal life to video games, politics, and wine. Because of their informal tone, blogs are also used by companies to improve relations with their customers. For example major anti-virus companies have blogs to discuss research on current threats [6]. Blog articles called *posts* are displayed in reverse chronological order: the latest post appears first. The Blogosphere is a collective term referring to all blogs and their interconnections, coined in 1999 by Brad L. Graham as a joke [8].

The blogosphere can be viewed as a graph, with blogs as nodes and edges corresponding to TrackBacks between blogs. A TrackBack produces a link from one blog post to another that references it. For instance if *Blog B* discussing French wine cites a post on *Blog A* about Bordeau wine, the TrackBack mechanism allows *Blog B* to notify *Blog A* about this citation. As a result of this notification, *Blog A* may then display a link back to *Blog B*. This mechanism was designed to help blog readers navigate from one post to other relevant posts.

Blogosphere ranking.

Since 2002, TrackBack relations between blog posts have become increasingly important because they are used by specialized search and ranking engines such as technorati [23]. Such specialized engines use a different ranking mechanism than standard web page search engines such as Google: to rank a blog, specialized engines count the number of blogs that point to it, stemming from posts over the last 6 months. The rationale behind this metric is twofold: First, information provided by blogs is timely and may become less relevant in a matter of months. Secondly, if two blogs address the same subject, it is likely that there will be many TrackBacks between them. Therefore counting the number of blogs is more informative than the number of links.

TrackBack.

As explained above, the TrackBack mechanism is important because it provides link reciprocity: "*When I cite you, you cite me*". The TrackBack mechanism operates automatically for two reasons. First, it is otherwise tedious and error-prone for a blogger to notify each blog cited in a post. Second, it would be time consuming for the recipient of manual notification to add links to all the blog posts that cite it.

The TrackBack specification was created by Six Apart, which first implemented it in its Movable Type blogging software in August 2002 [3] . Technically, the TrackBack mechanism is implemented in two parts: the **auto-discovery mechanism** and the **notification page**.

The **auto-discovery** mechanism uses a small RDF fragment added to each blog post that tells other blogs to which page they should submit their TrackBack. Inside the RDF fragment, the element dc:identifier contains the permalink of the blog entry, and the element trackback:ping contains the TrackBack URL, i.e. an URL to the **notification page**. An example is given below for illustration:

```
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-
          syntax-ns\#"
        xmlns:dc="http://purl.org/dc/elements/1.1/"
        xmlns:trackback="http://madskills.com/public/
          xml/rss/module/trackback/">
  <rdf:Description
        rdf:about="http://blog.foo.com/mypost"
        dc:identifier="http://blog.foo.com/mypost"
        dc:title="my post"
        trackback:ping="http://blog.foo.com/
          trackback/mypost" />
</rdf:RDF>
```

The **notification page** is the web page dedicated to collecting TrackBacks and processing them. Accordingly, the TrackBack ping itself is an HTTP POST request sent to the notification page. It contains four post values: the *post title*, its *URL*, an *excerpt*, and the *blog name*. An example TrackBack [3] post request is:

```
POST http://www.example.com/TrackBack/5
Content-Type: application/x-www-form-urlencoded
title=Foo&url=http://www.bar.com/&
excerpt=My+Excerpt&blog_name=Foo
```

There are other mechanisms, such as *Refback*, and *Pingback*, that provide similar functionality to the TrackBack mechanism. However they are not as popular as TrackBack, which is supported by every major blog platform except Blogger.

Blog spam.

Because TrackBack provides an automated way to insert links into other bloggers' blogs, it is not surprising that malicious users began using it soon after it appeared. There are two main motivations for abusing the TrackBack mechanism: search engine optimization, and spam to lure users to malicious sites. One of the major spam-blocker providers, Akismet [20], reported blocking around 15 million TrackBack spams a day in April 2009, in comparison with 1.8 million legitimate TrackBacks. Hence it seems that the ratio of ham/spam for blog spam is slightly better (90%) than the 98% spam reported for email [9]. However blog spam is more pernicious because it is asymmetric: one spam TrackBack might lure thousands of blog readers to a malicious site, whereas delivered email spam usually reaches at most one user. This asymmetry is very appealing to spammers, because a small quantity of TrackBack may potentially provide a large amount of traffic.

3. HONEYBLOG SETUP

In this section, we describe the experimental setup that allowed us to track spam activity. In order to observe blog spam activity, we ran a *"honeyblog"*, which was a real blog with real posts but with a modified TrackBack mechanism. Our experiment was conducted in two phases. The first phase was designed to establish our blog and attract real users. The second phase was an observation phase designed to track spam activity.

Setup phase.

In 2007, for a period of four months, we ran a real blog on information security with it own domain [1]. This blog was built on one of the most popular blog platforms at the time, dotclear [4]. During this setup period, we built our audience by posting frequently, gathering TrackBack and throughly referencing it. By the end of the setup period, Google analytics reported that we had more than 100 unique visitors per day. Predictably, TrackBack spam started to hit our blog after a few weeks. Once we observed multiple spam attempts per day, we replaced the dotclear TrackBack mechanism with our own code and began the observation phase.

[1]This blog is still active so we will not disclose its name.

Observation phase.

Our modified TrackBack mechanism was designed to accomplish two things: record TrackBack activity and deceive the spammers. Our custom TrackBack code stored every TrackBack submitted to the blog in a database. We recorded the data posted, HTTP headers such as the user agent, and the IP of the sender which we used later for geolocation. At some point, we tried to record more information by sending JavaScript tracking code to the spammer, but this was ineffective because spammers do not use browsers to send their spam. To learn more about the spamming platform, we also ran the passive OS fingerprinter p0f2 [25]. This passive fingerprinting allowed us to learn which OS the spammers use, the network distance to them in hops, and the uptime of their computers. Before we started the experiment, we expected that as for email spam, TrackBack spammers would use bots. As we will see in the next section, we were wrong.

One of our main concerns at the beginning of the experiment was to make sure we could distinguish real TrackBack from spam. A simple trick allowed us to do this. Recall from the previous section that the real TrackBack mechanism uses the discovery mechanism to determine where to send TrackBack. It turns out that our spammers did not use the discovery mechanism. Instead, they were simply using the default page URL. Therefore, we replaced the default notification page with a recording mechanism and modified the discovery mechanism to direct legitimate TrackBack to an alternate URL. Unfortunately, this method cannot be used as a spam defense because once the defense becomes known, spammers can easily adapt and use the discovery mechanism. To make the spammer oblivious to our modification, we modified the blog post display mechanism. For each visitor, our code looked in the spam database for spam from the same C IP class. If spam from the same approximate location was found, we displayed the last 50 entries to give the possible spammer the impression that their spam has not been blocked. We do not know if this deception mechanism was effective, as it is very hard to distinguish between a spammer crawler and the spammer itself.

We conducted our observations for about a year, between March 2007 and April 2008. During this time, we collected almost 10 million examples of TrackBack spam that were related to various "advertisement" campaigns. In the following sections, we focus on the campaign that accounted for more than 90% of the spam activity observed during our observation period.

4. SPAM PLATFORM

As depicted in *Fig.1*, we collected around 9 million blog spam entries from March 2007 to April 2008. The spam rate peaked in November 2007, with around 90,000 spams per day. Based on the number of unique IPs per day (*Fig.2*), we can partition the spam campaign into 3 phases.

The first phase, from March to early April 2007, appears to be the discovery phase for the spammers. This phase is characterized by a relatively low volume of spam, originating from multiple countries (*Fig.3*).

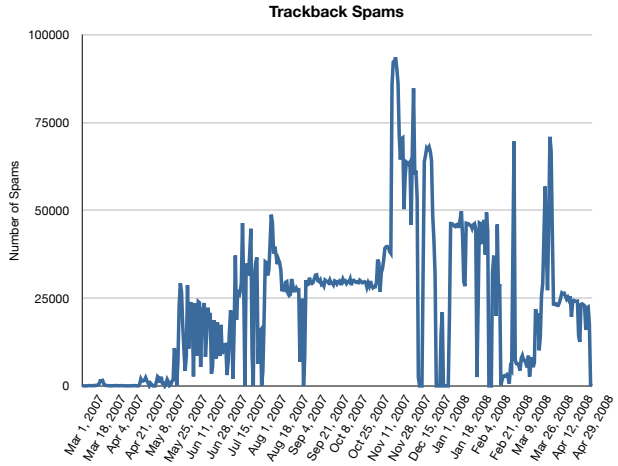

Figure 1: TrackBack Spams by Day

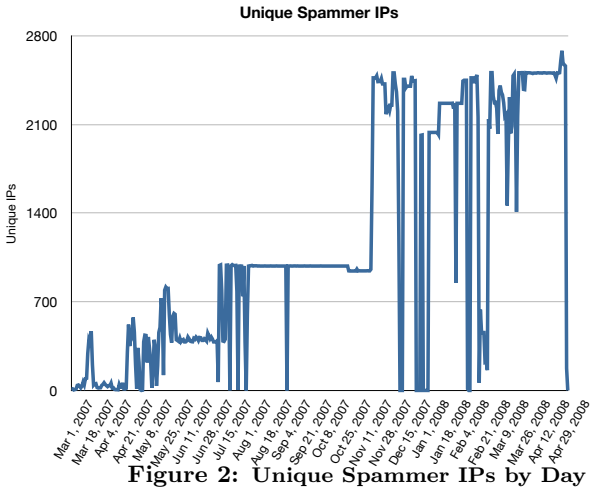

Figure 2: Unique Spammer IPs by Day

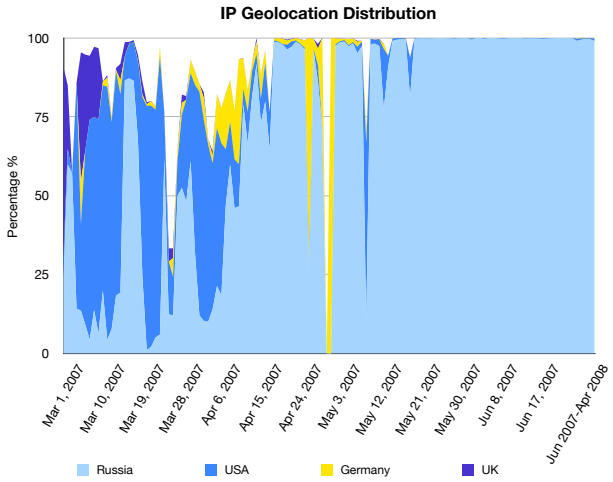

Figure 3: IP Geolocation Distribution

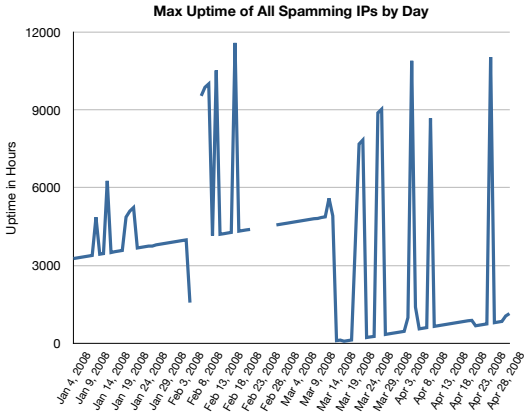

Figure 4: Maximum Uptime of Spamming Hosts

The second phase, from April to June 2007, was a period of small-scale spamming, with fewer than 700 unique IP addresses per day, predominantly from Russia. This suggests that this phase was used by spammers to see if their "business model" was viable.

Finally, the third and last phase of the spam campaign, from July 2007 until April 2008, was the most intense. During the final phase, we measured between 1000 to 2000 unique IP addresses per day, all from Russia.

The uptime of spamming hosts, determined by passive fingerprinting, suggests that the spammers used multiple IP addresses on each machine. In particular, we observed that an entire class C of IP addresses shared the same uptime for a period of four months.

As stated earlier, the most surprising result of our study is that the TrackBack spam we collected did not seem to originate from bots running on compromised end-user machines. Three observations led us to this conclusion: First, *Fig.4* shows that the spamming hosts were stable, and some had uptime as long as 1.5 years. This is uncommon among end-user machines, since they tend to be rebooted relatively frequently. Second, we plotted the average spam volume against the hour of the day and found that the spam rate stayed almost constant throughout the day. In contrast, we expect a bot running on a compromised end-user machine to show the effects of user activity patterns across the day. Third, our passive fingerprinting showed that 99.9% of spam originated from hosts running FreeBSD 5.3-5.4, which is not a common operating system for end-users.

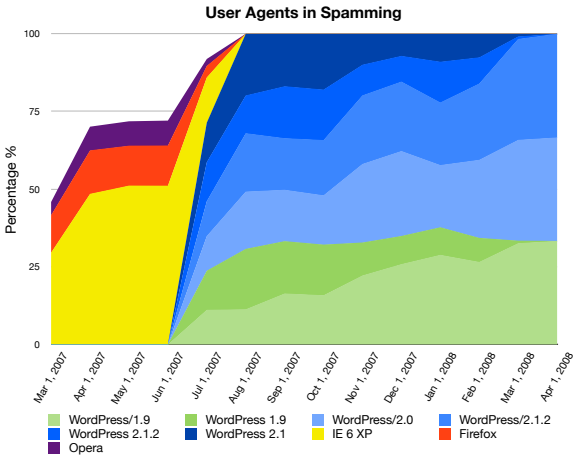

Figure 5: User Agents in Spamming

Our conclusion that the spam campaign started with a discovery phase is supported by the agent distribution presented in *Fig.5*. Because spam-generating software can report any user agent, it is interesting to see what user agent the spammers chose to report. At first, the spammers reported common browsers as user agents, which is anomalous because legitimate TrackBack pings are sent by blog engines. It was not until June 2007 that the spammers realized their mistake and started to report the popular WordPress blog platform as their user agent.

5. SPAM CONTENT

Figure 6: Top 100 Words in Blog Spam Excerpts

In this section, we analyze the content of the TrackBack spam received at the collection site. In total, 209,548 unique words were found in the spam excerpts sent in TrackBack pings, with an average of 5.8 words in each excerpt. As expected, the word distribution had a strong adult theme. The tag cloud in *Fig.6* was generated from the 100 most frequent words found in the TrackBack excerpts, using a free web application [7].

Table 1: A TrackBack Spam Example

Title	*Mighty morphin power rangers adult deluxe costume*
Excerpt	*costume Mighty morphin power rangers adult deluxe costume ...*
URL	*http://mighty-morphin-power-rangers-adult-deluxe-costume.samsbuy.nx.cn/index.html*
Blog_name	*Mighty morphin power rangers adult deluxe costume*

The word distribution in the titles, blog names, excerpts and URLs of the TrackBack spam were similar. In particular, the URL of each TrackBack spam was frequently formed mechanically from the same words found in the excerpt, with an example shown in *Table 1*. This pattern might serve as a discriminator to classify blog spam. However, it is likely that spammers will change this behavior as soon as they discover that it is used against them.

6. LURE SITE

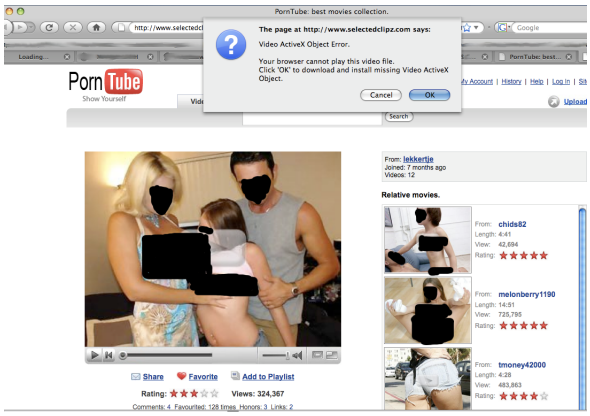

Figure 7: Blog spam lure site

In this section, we present our findings on the lure sites advertised by spammers.

All the URLs in the TrackBack spam we examined pointed to malicious web sites that tried to lure the user into downloading the malware *TrojanDownloader:Win32/Zlob.gen!dll*. We found that 97% of these URLs were used only once. The spammers primarily used sub-domains of the top-level domain *nx.cn*, with "appealing" keywords in the hostname, such as:

`http://adult-dating-in-altamont-tennessee.nx.cn`.

In every URL we tested, the lure was always based on the same idea: the web page displayed fake pseudo-pornographic video and asked the user to download an activeX control to view the video (Fig.7). Notice that the template and the name of the lure site were based on youtube. Of course, instead of supplying video, the web site only contained JavaScript and images. Each fake video provided a link to malware. The malware was not hosted on the lure site, but on a different website that was reached by multiple HTTP redirections.

The domain used to host the malware changed over time, but the listed owner of all domains was the same, *Igor Palki*. The registered address for all the malware sites was a small town in Russia, *Nijnii Novgorod*. An interesting fact about the malware website is that if one tried to access the site directly without supplying the proper URL, the site would not display anything. Even requesting the correct URL was not enough; the parameter *id* must be supplied to download malware. Interestingly enough, the malware binary changed according to the id supplied. Some of the domains used to host the malware were: *clipztube.com, vidztunnel.com, and vidzselector.com*.

The story behind the lure sites used by the spammer is also interesting. In the extensive spam campaign we identified, all the lure sites were associated with the same domain, *nx.cn*. In other words, although spam was posted from Russian IP addresses, the lure URLs came from a Chinese domain. It turns out that *nx.cn* is the official domain for the Ningxia province in China. The website www.nx.cn by itself is a social networking website for this province where users can create their own pages.

We found evidence in the nx.cn website suggesting that prior to April 2008, few security checks were used to restrict account creation. The introduction of Chinese CAPTCHAs subsequently seemed to account for the end of this particular spam campaign. Based on this evidence, CAPTCHAs appear to improve site security effectively by limiting automatic account creation.

7. TALKBACK : A PROPOSAL TO MITIGATE TRACKBACK ABUSE

Based on our study of TrackBack spam, we believe that a *secure TrackBack* mechanism with features described in this section can effectively mitigate TrackBack abuse. We plan to complete the design of the preliminary proposal outlined here and evaluate the resulting mechanism as a TrackBack defense.

We hope to develop and evaluate a new TrackBack protocol specification called **Talkback**, compatible with the current mechanism, that uses a trusted authority. This authority can provide a free lightweight public key infrastructure (PKI) and verify the identity and behavior of TrackBack senders and receivers to prevent TrackBack abuse.

Goals.

The planned TalkBack (secure TrackBack) mechanism aims at achieving the following goals:

- **Authenticated TrackBack**: We plan to authenticate the TrackBack sender and its blog by using a lightweight PKI. In order to do so our protocol requires that the blogger registers his blog before he can send TrackBacks. This registration is done in two steps. In the first step, the user registers to the authority so his account is linked to an email that we verify and a password. In the second step the user generates his public key and proves that he is the owner of his blog by putting a link to it in the headers of his blog. We also require that he adds in the headers a random number supplied by the authority. This random number is used to ensure that the user has not reclaimed a public key that he does not own. The registration process will use additional checks like CAPTCHAs to prevent automated registration. The security environment of the blog site, including the security of its authentication mechanism and how the blog platform behaves with multiple authors sharing the same blog, is an area we plan to study further.

- **Limiting TrackBack**: The second goal is to rate-limit the overall number of TrackBacks that a specific blog can send (to all receiving blogs implementing our mechanism). Determining a reasonable limit is part of our future work; we plan to compare negotiation protocols and policies allowing a blogger to request a higher limit, perhaps by solving a CAPTCHA or demonstrating some other form of reliability. Another option is to use blog ranking or age to give higher limits to blogs that are less likely to be controlled by a spammer.

- **Blacklisting abuser**: As explained in Sec 2, due to the asymmetric nature of TrackBack spam, limiting the overall number of TrackBacks may still allow dangerous blog misuse. Even if the spammer is able to send only a few spams to the most visited blogs, the spamming campaign may still have a huge impact. To address the threat of low-volume attacks with high impact, we propose a reporting mechanism that will report such abuse, presumably when identified by human readers of a blog. We also want to experiment with a propagation mechanism that will allow the authority to alert blogs proactively so they can reject unsent TrackBack pings by revoking the sender's public key. We envision the use of a reputation system to mitigate report abuse. This part of the system may use blog reputation and the number of distinct blogs that report the same sender. Our system will be able to defend against report abuse because the sender also has a certificate that can be revoked in case of misuse.

Protocol overview.

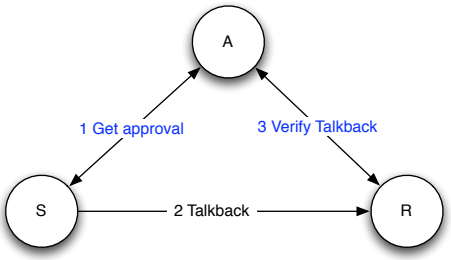

Figure 8: Talkback Overview

An overview of our envisioned protocol is depicted in *Fig. 8*. This diagram assumes that both the sender and the receiver have a secure TrackBack compatible blog and have registered with our central authority. This is a three step protocol:

1. **Getting a token**: Before a post is published, the blog contacts the authority to request a token. This token or a number derived from this token will be added to the TrackBack to ensure traceability and avoid replay attack. With this token the authority can inform the blog how many TrackBacks the sender is allowed to send using this token and when this token will expire. The expiration time is used to ensure that an attacker cannot accumulate tokens over time and then use them all at once to flood the blogosphere. If a blog reaches its TrackBack limit the server will require the blogger to visit a page to negotiate a higher limit by solving a CAPTCHA, for instance. The process of requesting a token from the authority can be transparent to the blogger, implemented in an iframe inside the blog interface. Our preliminary implementation work on a WordPress plugin suggests that this kind of integration can be achieved fairly easily.

2. **Posting the TrackBack**: Our planned TalkBack posts are very similar to the exisitng TrackBack specification. The sender sends a TalkBack to the receiver through a HTTP post. To be compatible with the TrackBack specification, this post request will contains the exact same four variables plus additional information, such as the signature over the post and the sender and receiver public keys. The receiver's public key will be retrieved by the sender via the discovery mechanism. This will ensure that an attacker cannot simply bombard arbitrary blogs but must contact the blog and fetch its key. Similarly, we advocate randomizing the notification page address to prevent DDOS attacks. The public key of the sender can be used by the receiving blog to create a whitelist once a TrackBack has been accepted. This provides a potential fallback mechanism in case the authority is offline.

3. **Verifying the TrackBack**: Finally the receiver queries the authority to determine whether the TrackBack is valid or not. The authority responds with the answer 'yes', 'no' or 'revoked'. The answer is 'no' when the sender quota has been exceeded. The answer is 'revoked' when the sender key has been revoked and should be blacklisted. In order to offload the network, we propose to allow batch submission and throttle the verification to one by minute.

While we have devised a protocol with a number of options, our next step is to test our protocol and implement it as a central authority and a WordPress plugin. We plan to take advantage of Stanford hosting capability to deploy the authority and provide this service for free to bloggers. Implementing and running the authority will allow us to collect feedback from the blogosphere and eventually formalize it into an open specification or a standard. Although the TalkBack posts above add signatures to each post, and impose additional communication overhead with the Talkback authority, we expect to investigate the trade-offs associated with various ways of reducing communication overhead by combining several logical messages in the same network communication.

8. ALTERNATIVE APPROACHES

TrackBack Validator. The WordPress TrackBack Validator [21] looks at the sender URL to validate that the post contains the URL of the receiver. This approach increases the network load because each receiver will look at the sender page. This load increase can be used to perform a DOS attack with amplification: the attacker spoofs a simple HTTP request and the receiver will fetch the entire page. With our central authority this problem does not exist: only the authority will verify that all the links exist.

Reputation system. Using a reputation system alone for TrackBack spam is ineffective because an attacker may change the blog URL for every posts. Therefore any long-term classification based on TrackBack is bound to fail because there is no way to prevent spoofing (under the current TrackBack specification).

IP Blacklisting. While blacklisting based on IP might currently work as the spammers today seem to use only a small number of IPs, it is not a sustainable solution because in the long run, it is likely that spammers will use botnets and therefore have a huge pool of IPs.

Rate limiting. Rate limiting at the blog level is not effective because a blog does not have a global view of the situation and therefore cannot stop spammers that target a huge number of blogs and post only once to each of them with the same IP.

9. ADDITIONAL RELATED WORK

Previous studies of spam email report that around 120 billions spam emails are sent every day [9]. In [10] and [13], the authors study a spam campaign by infiltrating the Storm botnet, while [1] analyzes the revenue generated by Storm spam. Former spammers relate their experiences in [11] and [24]. Blogosphere evolution is considered in a number of studies, including [14, 22, 16]. A DOS defense study [15] notes that ideas spread more quickly in the blogosphere than by email. In previous work on linkback spam, [18] examines ways that the language appearing in a blog can be used as a blocking defense. Similarly, [19] studies how the language of web pages, including blogs, can be used to detect spam. In [12] the authors use Support Vector Machines (SVM) to classify blog spam.

10. CONCLUSION

In this paper, we describe a longitudinal study of TrackBack blog spam and propose improvements to the TrackBack mechanism that prevent its misuse. In the longitudinal study, which may be the first publicly reported, we instrumented an operating blog to collect nine million examples of TrackBack spam. Over 90% of the collected TrackBack spam came from one extensive spam campaign, which we further analyzed. Based on the data we collected, we conclude that the TrackBack spam was generated from a relatively small number of stable, long-lived hosts with a large number of different IP addresses. This stands in contrast to common email spam campaigns through bots on compromised end-user machines. One possible explanation for the difference is that it is currently easier to spam blogs than email addresses because fewer defenses are currently in place. The TrackBack spam we collected was sent from Russian IP addresses, and

directed blog readers to lure sites at Chinese domains that asked users to download malware as an "ActiveX control" needed to view advertised video. This large TrackBack spam campaign apparently ended when Chinese CAPTCHAs were installed, restricting the spammers' easy control of a large number of Chinese domain names. We outlined the design of a global awareness system that will allow us to obtain a better understanding of blog spam activity, and develop future TrackBack spam defenses based on our observations.

11. REFERENCES

[1] Clive Akass. Storm worm 'making millions a day'. http://www.pcw.co.uk/personal-computer-world/news/2209293/strom-worm-making-millions-day, Feb 2008.

[2] Six Apart. Six apart guide to comment spam. http://www.sixapart.com/pronet/comment_spam.

[3] Six Apart. Trackback technical specification. http://www.sixapart.com/pronet/docs/trackback_spec.

[4] Dotclear. Dotclear blog plateform. http://dotclear.org/.

[5] Tom Espiner. Filipino news site hit by trackback spam. ZDNet Asia, http://www.zdnetasia.com/news/security/0,39044215,61998878,00.htm, 2007.

[6] F-secure. F-secure blog. http://www.f-secure.com/weblog/.

[7] Jonathan Feinberg. Wordle. http://www.wordle.net/.

[8] Brad L. Graham. Bradland must see http comments. blog http://www.bradlands.com/weblog/comments/september_10_1999/, Sep. 1999.

[9] Ironport. Internet security trends. http://www.ironport.com/securitytrends, 2008.

[10] Chris Kanich, Christian Kreibich, Kirill Levchenko, Brandon Enright, Geoffrey M. Voelker, Vern Paxson, and Stefan Savage. Spamalytics: an empirical analysis of spam marketing conversion. In *CCS '08: Proceedings of the 15th ACM conference on Computer and communications security*, pages 3–14, New York, NY, USA, 2008. ACM.

[11] J. Kirk. Former spammer: 'I know I'm going to hell'. http://www.macworld.com/article/58997/2007/07/spammer.html, July 2007.

[12] Pranam Kolari, Akshay Java, Tim Finin, Tim Oates, and Anupam Joshi. Detecting spam blogs: A machine learning approach. In *2006. Proceedings of the 21st National Conference on Artificial Intelligence (AAAI)*, 2006.

[13] C. Kreibich, C. Kanich, K. Levchenko, B. Enright, G. Voelker, V. Paxson, and S. Savage. Spamcraft: An inside look at spam campaign orchestration. In USENIX, editor, *LEET*, 2009.

[14] Craig Macdonald and Iadh Ounis. The trec blogs06 collection : Creating and analysing a blog test collection. *DCS Technical Report Series*, 2006.

[15] Ashraf Matrawy, Anil Somayaji, and P. C. Oorschot. Mitigating network denial-of-service through diversity-based traffic management. In *ACNS'05*, pages 104–121. Springer Science+Business Media, 2005.

[16] Universal McCann. Power to the people - social media tracker wave.3. http://www.universalmccann.com/Assets/wave_3_20080403093750.pdf.

[17] Declan McCullagh and Anne Broache. Blogs turn 10–who's the father? http://news.cnet.com/2100-1025_3-6168681.html.

[18] Gilad Mishne, David Carmel, and Ronny Lempel. Blocking blog spam with language model disagreement. In *In Proceedings of the First International Workshop on Adversarial Information Retrieval on the Web (AIRWeb)*, 2005.

[19] Alexandros Ntoulas and Mark Manasse. Detecting spam web pages through content analysis. In *In Proceedings of the World Wide Web conference*, pages 83–92. ACM Press, 2006.

[20] Automattic Production. Askimet trackback statistics. http://akismet.com/stats/.

[21] Dan Sandler and Andy Thomas. Trackback validator. http://seclab.cs.rice.edu/proj/trackback/.

[22] Technorati. State of the blogosphere. http://technorati.com/blogging/state-of-the-blogosphere/.

[23] Technorati. Technorati top 100 blogs. http://technorati.com/pop/blogs/.

[24] D. Watson. All spammers go to hell (posting to funsec list). http://www.mail-archive.com/funsec@linuxbox.org/msg03346.html, July 2007.

[25] Michal Zalewski. P0f2: "Dr. Jekyll had something to Hyde" passive OS fingerprinting tool. Web, 2006.