# BlogSpamAssassin

## Blog Spam

Blog spam is completely different from e-mail spam. The objective of the e-mail spammer is for you to read their message and respond quickly. The opposite holds true of the weblog spammer. The spammer needs their comments to remain undetected (or at least undeleted) to boost and maintain the pagerank of the site that they are spamming for. This type of spam is not limited to blogging systems and can easily be expanded to take into account other collaborative portals (e.g., wiki, forums, etc.). In regards to blogs, the main ports of entry for spam are:

- Comments (especially from anonymous users)
- Trackbacks
- Pingbacks

The last of these two do not require any human interaction at all and are more automated processes of communication. While Pingbacks require links back to the system being commented on, they too can be spoofed. While many systems have anti-spam measures on the web interface to prevent automated comment spam (e.g., Captcha, arithmetic or logic questions, obfuscated javascript code, etc.), the main concern of this article is for processing spam that gets beyond the UI.

## SpamAssassin Integration

While there are difference between e-mail spam and blog spam, SpamAssassin is a strong candidate as a basis for preventing blog spam. There have already been several attempts to integrate SpamAssassin with a blog (WordPress and Moveable Type):

- http://apthorpe.cynistar.net/code/babycart/
- http://www.hjackson.org/blog/archives/2004/11/moveable_type_s.html
- http://www.kahunaburger.com/2005/01/11/spam-assassin-and-movable-type/

These plugins basically take the content from a blog, tests it with SpamAssassin, and flags it as needing moderation if deemed unsafe.

## Miscellaneous Notes

- Proof-of-work: A legitimate user will take several seconds to minutes to create each unqiue comment while a comment spammer sends them out as fast as possible. Consider a proof-of-work algorithm executed within the browser (e.g. javascript, java, activex) added to comment submission forms. The weblog software can safely reject all comment submissions that lack valid proof of work. Legitimate users will not be inconvenienced by a short delay as they submit their comment while spammers will not be able to easily submit comments in large volumes. For example, if a typical comment spammer sends 1000000 comments per day and the proof of work requires 2 seconds of compute time then they will need to dedicate 24 machines to proof-of-work computation to maintain their rate of transmission. The cons of this method are that users without advanced browsers or older, slow computers may not be able to post comments.
- Collaborative filtering: IronPort maintains a database of e-mail server traffic volumes called SenderBase. Mail servers can use SenderBase to find "traffic spikes" and potentially block e-mail from those servers. Something similar could be done for weblogs. As comments come in, weblogs could report the urls in the comments to a central server. If an URL is sent in too rapidly, it can be added to a list of probable spam urls and weblogs can quarantine or delete comments containing that url.
- DNS-based URI Blocklists: SpamAssassin has had great success using Jeff Chan's Spam URI Realtime Blocklists. When an e-mail arrives, Spam Assassin extracts the urls contained within and performs a few DNS TXT queries to find whether the url has been reported in spam. These blocklists can be used for weblogs too. Instead of Jay maintaining a central blocklist that people download and install manually, mt-blacklist could use a DNS-based blocklist that is effectively updated in real time. This would significantly cut down on comment spam because weblog owners would not need to actively maintain their blocklists. The submission process could be streamlined so that it doesn't consume so much of any one person's time.

## Other Resources

- WordPress article
- Drupal article
- Three-part article on detecting comment spam

### PHSDL

Sharing its methodology with BlogSpamAssassin under PHSDL GNU.

- PHSDL http://www.phsdl.net
- PHSDL Malware and Redirect Spam Domains Live Public List
- BlogSpamAssassin Mailing List Contribution by PHSDL
- PHSDL Honeypot Forum

### Project Framework Constraints

- Not to stop search engine Spam
- Stop comments Malware and redirect domains Spam
- Not to stop off topic comments Spam

- Develop a universal API anti Spam filter for different Scripting languages
- API anti Spam Filter must be available for forums, blogs, and book marking services

## BlogSpamAssassin Directives

- PHSDL is supported by StopBadware
- StopBadware uses Malware API list from Google
- BlogSpamAssassin can adopt the StobBadware Google API framework
- API Anti Spam Filter can be developed for WordPress, phpBB, vBulletin, and SMF

## Avoid Problemetic Honeypots

- WikiPedia domain is not Spam (Open relay error)
    - http://www.projecthoneypot.org/comment_spammer_urls.php
- False Positives Documented (Human editor error)
    - http://www.phsdl.net/phsdl-vs-akismet-complaint.php

## Minimize resource consumption

- Structure SBL by domain not by url or sub domain
- Query SBL primary database for domain match
- Utilize push technology vs. pull technology

## Adopt Standarized Filtering Techniques

- Primary Filter needs to be based on Malware and cloaking redirect Domains
- Secondary Filter should be based on ParentProject SpamAssassin
- Tertiary should be user defined white and black list Filters

## BlogSpamAssassin Sub Algorithm PHSDL Filter Test

- Select a Malware domain from PHSDL Malware Domains Public List
- Post To NoSpam PHSDL Blog

# Mailing List

A mailing list has been created to begin work on this project.

blogspam@spamassassin.apache.org

You can subscribe via:

blogspam-subscribe@spamassassin.apache.org