The problems associated with operating an effective anti-spam "blocklist" system in an increasingly hostile environment.

Robert Gallagher
MSc in Security and Forensic Computing - School of Computing, Dublin City University.

August 30, 2004

Abstract

Unsolicited Bulk email, commonly referred to as 'Spam', is a problem that has received widespread attention in the media and academic circles. Innovative technical solutions have been proposed, but such solutions would be very difficult to implement on top of the current email system because of its widespread deployment. Legislative measures have been put in place but these have not taken the decentralised nature of the Internet into account and often assume cooperation from the spammers (section 2.2.4).

Anti-spam 'blocklist' sites take a more direct approach by providing email users and Internet Service Providers with databases of Internet hosts (IP addresses) known to harbour spammers. In this manner blocklists have had much success in reducing the overall level of spam. However, as bulk emailers become more sophisticated in their techniques these blocklist sites are falling under frequent attack.

This practicum will investigate the problems faced by blocklist systems through the techniques employed by spammers and outline a possible solution based around a distributed blocklist system.

1 Introduction

The earliest known email that could be classified as spam was sent in April 1994 by the law firm Canter & Siegel, advertising their services to those wishing to take part in the US government lottery of green card work permits [1]. Around six thousand copies of this message were posted to Usenet discussion forums, breaching the unwritten rules of 'netiquette' that had governed behavior on the newsgroups up until that time.

The terms 'spamming' and 'spam' had been coined to describe widespread and unwanted postings to Usenet newsgroups, which would usually be unrelated to the topic being discussed. The term spam is a reference to a Monty Python sketch in which spam is the main ingredient of every dish in a café.

1.1 The Real Cost of Spam

Ten years later, in April 2004, spam accounted for 67.6% of 840 million messages assessed by the security firm MessageLabs [2]. The combination of an enormous potential audience and the ease of reaching that audience has made email a very attractive medium for marketing, scams, politics and religion. Legitimate businesses and users have paid the price however. In a study conducted by the Radicati Group [3], it is estimated that deploying extra infrastructure to deal with spam cost companies around the world 16.7 billion euro (20.5 billion US dollars stated in the report) in 2003, and this is set to rise to well over 60 billion euro by 2007 (figure 1).

1.2 Simple to Sophisticated

Attempts have been made to reduce spam both at the client and server levels, from simple keyword filters to bayesian filters and blocklists - examples of which are described in section 2.1. These techniques have grown more sophisticated

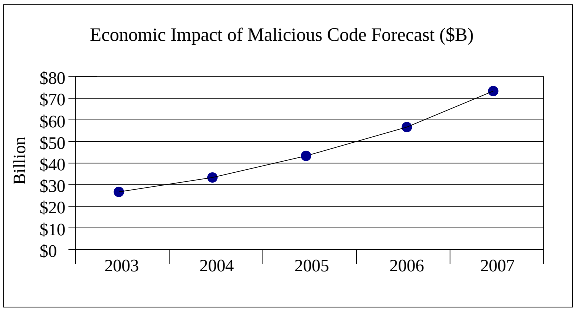

Figure 1: Economic impact of spam and malicious code, 2003-2007 (billions).

as the volume of spam has increased, but unfortunately the tactics employed by spammers have become just as ingenious. This has led to what some have termed the spam 'arms race' [4].

Spammers have begun to enlist the services of malware[1] authors in order to create viruses and worms that aid in the distribution of spam, usually with the purpose of concealing its origin. In section 3 the growing links between spammers and malware authors are illustrated, which is one of the main concerns of this practicum.

2 Blocklists

Blocklists are simply databases that contain the IP addresses of known spam operations or computer systems that can be exploited to send spam. Most modern SMTP servers can be configured to query a blocklist on receipt of an email message [5], extensible filters such as SpamAssasin can also make use of blocklists.

If the source IP of the message (retrieved from the email headers) exists in the database, the server can discard the message altogether or flag it as possible spam - in this case it is up to the client side email application to deal with the message.

ISPs, educational institutions, businesses and government agencies have all made extensive use of blocklists. Many blocklist systems have come into being since the earliest, MAPS RBL (section 2.1.3), began operation in 1996. Some of these systems are free, others are partly subscription based providing extra services, more comprehensive filtering and support. Because of their popularity, many blocklists have become the target of attacks. These attacks are described in section 3.

The actual inclusion of an IP in a blocklist usually requires some human intervention in the form of a review process. Most blocklists encourage members of the public to submit spam sources through a well defined procedure, this minimises the number of false positives[2] that appear in the blocklist.

2.1 Current Blocklists

2.1.1 Spamhaus

The spamhaus project is one the better known blocklist systems, providing several core services. SBL (Spamhaus Block List) is a real time database of IP addresses associated with known sources of spam. Email servers can easily be configured to query the SBL on receipt of a message, and discard it if it comes from a verified spam source.

XBL (Exploits Block List) is similar to SBL, except it stores the IP addresses of 3rd party exploits such as open proxies and malware designed to aid in the distribution of spam.

ROKSO (Register Of Known Spam Operators) is a database that stores information and evidence on known spam operations. The information in ROKSO can be useful in tracking spammers activities, in particular any ISPs that they might be using to host their operations.

Spamhaus' widespread use by ISPs and other organisations led to it falling under successive dDoS (section 3.1.3) attacks throughout 2003 by the Mimail, Fizzer and SoBig worms. This, and other attacks against blocklists are described in section 3.

2.1.2 Spamcop

Spamcop began as a spam notification and reporting system. Emails reported to SpamCop are analysed to determine who originally sent them

[1] An umbrella term for computer viruses, worms and trojans.

[2] Legitimate servers wrongly listed as sources of spam.

and any email addresses or URLs in the body of the mail are recorded. The SpamCop system then contacts the relevant system administrators to inform them about the problem.

The reporting service quickly gained popularity and SpamCop began to offer commercial email accounts, site-wide corporate filtering and a blocklist service which solicits donations. However the blocklist that SpamCop operates has not been very successful [6]. The listing process that the SpamCop blocklist employs appears to result in large numbers of legitimate IPs being incorrectly listed.

2.1.3 MAPS RBL

The MAPS RBL[3] is a commercial blocklist that began operation in 1996, making it one of the earliest anti-spam systems [7]. Comprehensive guidelines have been formulated in regard to how sources of spam are to be reported, and what constitutes a spam source. The procedures used by MAPS RBL have been held up as an example of how reporting of a suspected spam source should be carried out [8]. MAPS offers several other IP address listing services that do not necessarily list known sources of spam, but poorly configured systems that could be used to send spam[4].

2.2 Alternatives to Blocklists

Blocklists have often been criticised for blocking legitimate email servers and being extremely slow to correct the error [9]. A lack of any accountable standards body, such as ICANN that regulates DNS, has compounded the problem. Whilst the idea of blocklists is a sound one, many see them as untrustworthy and over zealous. However, many alternatives to blocklists are available.

2.2.1 Content Based Filtering

The actual content of the email itself can be analysed to determine if it is a legitimate message or not. In fact early spam filters using hand crafted rules, such as regular expressions, were quite effective [10]. Users of newsgroups and mailing lists often employed content filtering to classify mails according to keywords in the subject line or the senders address, the mails would then be sorted into folders based on this. When spam first began to appear on newsgroups and in email, content based filtering was the natural choice to combat it. Because spam emails often had characteristic words and phrases it was a simple matter to adapt existing rules to move spam into special folders or delete it entirely. But as spammers grew more sophisticated simple filtering using keywords became less effective.

This forced content based filtering to evolve, using machine learning (ML) techniques to automatically classify messages. The Naive Bayes method has become the focus of much research and development involving ML-based spam filtering because of its superior ability to classify text [11]. Naive Bayesian filters recognise emails that are similar to a training set of messages, over time the filter becomes more accurate at classifying messages. Naive Bayesian filtering has been implemented in client-side email applications such as Mozilla Thunderbird [12] and server-side filters like SpamAssasin [13].

2.2.2 Fake open relays

Spammers are often attracted to open relays because they offer the possibility of relaying email in an anonymous manner. Modern SMTP servers will not relay mail by default and it is acknowledged best practice for system administrators not to configure them as such, however there are still vast numbers of poorly configured or un-patched servers connected to the Internet. It is worth the spammers while to expend large amounts of time and effort to locate these mis-configured servers. Projects such as spamhole.net [14] create networks of servers that masquerade as open relays, but in reality the message goes nowhere.

[3]Mail Abuse Prevention System - Realtime Blackhole List

[4]Open relays or open proxies

2.2.3 Message Signatures

A message digest of a known spam email is created and published in a directory. Filters such as SpamAssasin can then query this directory and flag as spam any messages that hash to digests present in the directory. Since spam emails are often duplicated this has proven to be quite an effective technique. The Razor [15] project implements this concept; users submit messages along with their one way hashes. Consistent successful reporting of known spam gives a user a higher rating of trustworthiness, meaning any spam they report in future will receive a higher priority for publishing in the directory.

2.2.4 Non-technical solutions

Non-technical solutions have mainly consisted of the formulation of new legislation or revising existing laws to make provisions for unsolicited bulk email. These measures have had little or no effect because, being confined to a single country or administrative region they fail to take into account the decentralised nature of the Internet. A spammer or spam gang[5] can easily reside in one country and host their email servers in a country with less-stringent legislation.

The EU Directive on Privacy and Electronic Communications [16] has attempted to make it illegal for any marketing information to be sent to an individual without their prior consent. Most member states, including Ireland, have adopted the directive but the EU has been slow to take action against countries that failed to incorporate it into their own laws. After the deadline of 31st of October 2003, eight countries had not yet adopted the directive [17].

In the US, the most widely publicised piece of anti-spam legislation has been the Controlling the Assault of Non-Solicited Pornography and Marketing Act of 2003, or the CAN-SPAM act [18]. This act requires all marketing information sent by email to include legitimate return addresses and instructions on how to opt-out of the mailing list. But lawyers have claimed that the act cannot be enforced in a practical manner and, more seriously, that it supercedes stricter state laws that give members of the public the power to sue spammers [19]. Because of its perceived leniency towards spammers, this act has often been referred to as the YOU-CAN-SPAM [20] act.

International cooperation and common legislation appears to be the way forward for effective anti-spam laws. In July 2004 the USA, UK and Australia signed a "Memorandum of Understanding" [21] that will allow governmental agencies in the three countries to share evidence against spammers and coordinate their enforcement efforts. The United Nations and the International Telecommunications Union have also indicated [22] that they aim to standardise anti-spam legislation around the world in the next two years.

3 Spam and Malware

During November 2003 the servers hosting the Spamhaus blocklist began receiving huge volumes of fabricated requests as part of a Distributed Denial of Service (dDoS) attack. The attack was launched from thousands of computers worldwide, that had been infected with the Mimail virus [23].

This incident was just one in an increasing number of attacks launched using malware that infects machines with the purpose of using them as 'zombies' for sending spam or conducting dDoS attacks against anti-spam organisations.

It is claimed that throughout August and September 2003, sustained dDoS attacks launched using malware caused at least three anti-spam systems to cease operations indefinitely [24]. The increasing sophistication of these attacks has highlighted a growing connection between spammers and malware authors.

3.1 Techniques and Tools

Today, spammers commonly employ 'Mass Mailer' worms to aid them in their activities. Mass Mailers are so called because they propagate by harvesting large amounts of email addresses from the target system to send copies of themselves

[5] Spam operations consisting of a large number of professional spammers.

to. Mass mailers are commonly designed to take advantage of Microsoft Outlook and Outlook Express. Since these email clients are in widespread use the worm will have more chance of success.

But worms have begun to emerge that have their own built-in SMTP engine (section 3.1.4), this allows the worm to send itself regardless of the email client being used, all that is required is TCP/IP port 25 to be accessible. The worm establishes a connection with an SMTP server (a remote server, or one that is part of the worm itself) that allows e-mails to be sent without verifying who is sending them or from where. This is possible because the SMTP protocol was designed long before the growth of the Internet, viruses and spam. As a result it is extremely permissive - any information at all can be entered into header fields, allowing the true source of a message to be effectively hidden [25]. Once established on target systems, spammers can use these worms for a wide range of activities; the most common being spam relaying, content hosting and denial of service attacks.

3.1.1 Spam Relays

Worms such as SoBig, Migmaf and Fizzer (section 3.1.4) install SMTP relay components onto the victim machine, allowing it to act as a proxy for large amounts of spam. In June 2004 the Network Management firm Sandvine determined that 80% of spam originated from zombie machines infected with trojans and worms [26], indicating an increasing tendency for spammers to use zombies as their preferred method of delivery.

3.1.2 Content Hosting

The worm can have its own built-in HTTP server for hosting websites that the spammer advertises in his emails. Such content is often illegal so it is in the spammers' best interest to host it somewhere that allows him to remain anonymous and, if there are a large number of zombie machines involved, the website is almost impossible to shut down.

In the case of the Migmaf Trojan, the zombie machine acts as a reverse proxy for a master server hosting the actual content [27]. When a request is received for a web page, it is relayed to the master server through one of the infected machines. The master server then sends the page back along the same chain to the user that requested it. Thus, the spammer is able to host his content with possibly legitimate providers and effectively mask its true location.

3.1.3 Denial of Service (DoS) Attacks

In recent years, network attacks have been characterised by 'Denial of Service', or DoS attacks. This takes the form of flooding target computers and networks with traffic with the intention of degrading performance or disabling the system completely. DoS attacks can be categorised as *single-source* attacks involving one host or *multi-source* attacks with two or more hosts flooding the intended target with attack traffic [28].

The simplest type of single-source DoS attack is a Ping flood. The Ping tool is useful for determining whether a system is properly connected to a network, and is available by default on most operating systems. It uses a form of data called Internet Control Message Protocol (ICMP) to send packets to a remote machine that sends a ping reply back acknowledging the request. Unfortunately, ping can also be used as part of a Denial of Service attack to 'flood' the intended target with multiple ping requests (ICMP packets) which cause the server to send back replies, resulting in network slowdowns and even crashes. A common technique is to spoof a source address for a large number of ping requests - the spoofed address being the target machine, the corresponding ping replies then overwhelm the target with no effect on the attacker.

Multi-source attacks are also referred to as distributed denial of service (dDoS) attacks. The dDoS has quickly become the weapon of choice for attacks against blocklists and other anti spam systems. Whilst ping floods using spoofed source addresses can be an effective means of disabling a target system, there is still the possibility that the attacker can be traced since he must initiate the attack and send the ping request packets himself. Because of this, many dDoS attacks are now carried out using ordinary home users machines infected with malware, effectively masking the originators identity and giving the attack a greater chance of success because of the large number of hosts involved.

3.1.4 Bringing it all together : The Fizzer Worm

An extremely sophisticated example that provides all of the above 'features' can be found in the Fizzer worm which, along with SoBig and Mimail, was responsible for many of the attacks noted in section 3. Fizzer spreads by emailing copies of itself to randomly generated email addresses and addresses found in the Windows or Outlook address books. The worm also disguises itself as a music or video file in order to spread through the peer to peer file sharing network Kazaa.

Its payload consists of installing a web server for hosting the spammers content, an IRC (Internet Relay Chat) backdoor, an SMTP engine and DoS attack tools onto the victim machine. The worm then waits for instructions to be sent to it through the IRC backdoor. In this manner Fizzer can remain dormant and undetected on a victim machine, until it receives instructions to activate.

4 Distributed Blocklist

Taking into consideration the techniques used by spammers, as described in section 3.1, blocklists operating from a single host or a small core of servers are increasingly vulnerable to attack from determined groups or individuals with powerful and easy to use tools at their disposal who have a lot to gain for comparatively little effort.

One approach to this problem is to make such attacks extremely difficult to mount effectively, such that the effort involved in carrying them out is significantly greater than the reward to be gained. This can be achieved by distributing the blocklist data over a large number of disparate peers or nodes.

At its heart a distributed blocklist is simply a system for storing data, along the lines of the popular Freenet [29], but with stricter controls over the integrity of the data. The data that is being distributed is a list of IP addresses for known sources (SMTP servers) of spam. Data is stored according to the block of IP addresses it describes. For example, we would have a section of the blocklist that would store any listed IP addresses in the 194.145.*.* range. Any queries for addresses in this range could then be immediately directed to that section of the blocklist. These sections can be referred to as *netblock sections* and they are the basic unit of data for the system. Storing data in this manner speeds up the execution of queries because only the netblock section in question in searched, not the entire list.

4.1 Desired Features

The following features would be desirable in a distributed blocklist system in order to make it an practical alternative to current blocklists that is resistant to the types of attacks described in section 3.1.

4.1.1 Trust

Perhaps the most important feature in the blocklist is that the peers can trust the data they receive. Many blocklists have failed in the past (section 2.2) because of a perceived lack of trust. Trust is doubly important in a distributed blocklist where the system consists of many unknown elements.

For this reason the design of the distributed blocklist incorporates *trusted maintainers* (section 4.2.1). A trusted maintainer entity makes its public key available to the peers who can then use it to verify any blocklist data that they receive.

4.1.2 Ease of participation

In order to have as many peers as possible, it should be trivial for any entity to participate in the blocklist. This can be accomplished by requiring a minimum amount of software on the client side and distributing the list over a commonly used protocol such as HTTP.

4.1.3 Caching

Queries for the same IP address may be repeated many times, so caching the results of queries locally improves efficiency by minimising such repeated queries and moving data physically closer to where it most requested. This is commonly referred to as "Edge of Internet Caching", or co-operative caching [30].

4.1.4 Integrity

The system should be resistant to poisoning attacks - corruption of the list by injection of false data. As noted in section 4.1.1, the system does consist of many unknown peers and it must be presumed that any of these peers are untrustworthy and may attempt to corrupt the system. The trusted maintainers are key to this requirement.

4.1.5 Robustness

The robustness of the system in this case would be its resistance to DoS/dDoS attacks, it should be extremely difficult to significantly affect or degrade this system. Unfortunately, no effective method exists of preventing a sufficiently determined party from launching a DoS or dDoS attack [31].

The trusted maintainers are easily visible targets and given enough resources an attacker may disable a large proportion of them. However, the list would still exist and be accessible since it is stored on the peer nodes. The only noticeable effects of a successful dDoS attack would be the loss of updates to the list and inability for new peers to join until the trusted maintainers are brought back online.

4.2 Design

The main activities that would be carried out by the entities in a distributed blocklist system would be *querying the list*, *joining the system* and *maintenance of the list*. Before these activities are outlined however, it is important to identify the entities that will participate in the distributed blocklist.

4.2.1 Trusted Maintainers

These are trustworthy entities that make decisions about what IPs to list and allow new peers to join the system (secion 4.2.4). The trusted maintainers may be blocklist operators that exist today, or well known organisations that already offer trust-based services such as Certification Authorities. Trusted maintainers could be listed in a public directory to allow them to be easily located by new peers wishing to join the system.

4.2.2 Peers

Peers form the backbone of the system by storing the blocklist data. Each peer has two data stores; a routing table and a cache. The routing table keeps track of other peers in the system that queries can be directed to and the cache stores blocklist data and the results of any successful queries. Data in the cache is not static however, netblock sections are deleted after a predefined amount of time in order to facilitate circulation of updated data. Also, newer versions of netblock sections received from the trusted maintainers overwrite older ones.

4.2.3 Querying the List

The following simple algorithm details the steps that are taken to check if a specific IP is stored in the blocklist (figure 2). For example, we wish to determine if 194.145.128.7 is listed, so we will request the 194.145.*.* netblock section.

Firstly, we check the required netblock section is not already in the cache. If it is not, we check another participant for an answer, this query may be referred until an answer is received - ie: the IP in question is listed, or not listed. If a successful answer is received it is verified using the maintainer's public key and then stored in the cache. To stop queries from circulating indefinitely, a *hops-to-live* value can be associated with each query message. This value is decremented by each peer upon receipt of the query message; the peer that receives a query message with a hops-to-live value of zero will not retransmit that message.

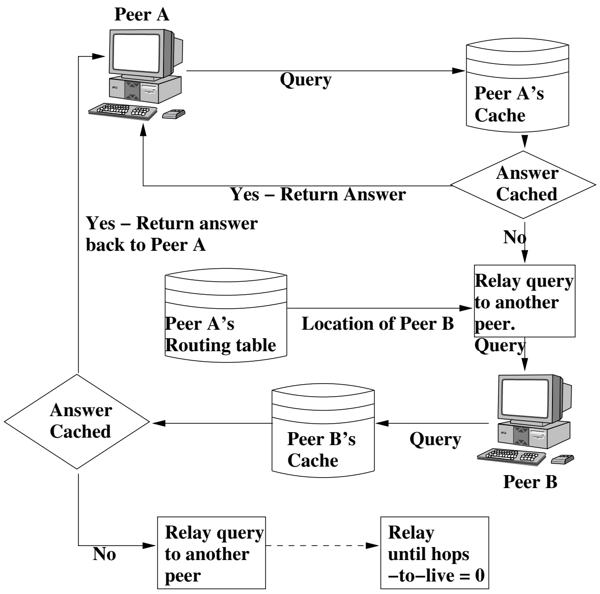

Figure 2: Query flow

4.2.4 Joining the System

Peer A announces itself to a maintainer server by sending its location. Peer A is then given a netblock section to store along with the location of another peer (Peer B) in the network and the maintainers public key.

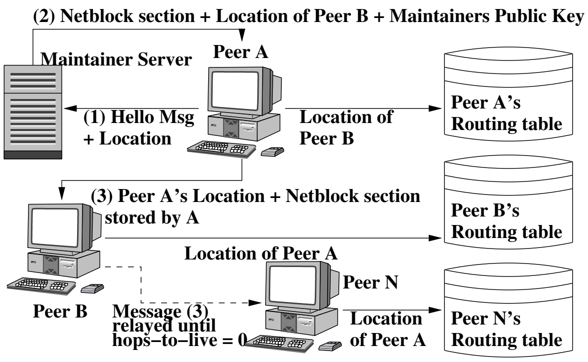

Figure 3: Adding a new peer to the system

Peer A then announces itself to Peer B. The message tells Peer B Peer A's location and what netblock section it is storing. Peer B then adds this information to its routing table. Again, a hops-to-live value could be associated with each message in order to stop it from being transmitted infinitely, but to allow the optimum number of peers to be aware of the new peer. The process of adding a new node to the list is shown in figure 3.

4.2.5 Maintenance

Maintenance of the blocklist largely consists of determining what IP addresses to add to the list and in some cases, removal of IP addresses where it has been sufficiently justified. Existing review processes such as those described in section 2.1.3 could be used to manage the blocklist. The trusted maintainers then release the updated netblock sections, signed with their private key, to several chosen peers. These updates propagate because any newer data will overwrite older data in the peers cache and 'stale' data is removed automatically after a certain amount of time (section 4.2.2).

5 Conclusions and Future Work

This paper investigated the increasing cooperation between spammers and malware authors and the threat this poses to current blocklist systems ability to operate effectively in the future. A solution was described that involved a distributed blocklist operating in a peer to peer fashion over the Internet. Storing the blocklist data in this manner would mitigate the effects of dDoS attacks since the accessibility of the blocklist does not depend on any particular component.

Development of the distributed blocklist introduced in this paper would have to involve a large number of peers. Since projects such as distributed.net [32] have set a precedent for large-scale distributed systems operating effectively over the Internet, sufficient interest could be gathered to allow a network to be quickly deployed. The distributed blocklist would easily integrate with current blocklist systems because it is simply a framework for the storage and retrieval of blocklist data in a distributed manner.

The sophistication of modern viruses and techniques employed by spammers means that blocklists must evolve to incorporate systems such as the distributed blocklist, if they are to remain a viable means of filtering spam in the future.

References

[1] Brad Templeton. Origin of the term "spam" to mean net abuse. *Essays on Junk E-mail (Spam)*, 2003. Article available at http://www.templetons.com/brad/spamterm.html.

[2] John Leyden. Two thirds of emails now spam: official. *The Register*, 2004. MessageLabs report cited in article on The Register - http://www.theregister.co.uk/2004/05/25/spam_deluge/.

[3] The Radicati Group. Anti-virus, anti-spam and content filtering market trends, 2003-2007, 2003. Cited in article: Spam will cost business $20.5bn this year - http://www.vnunet.com/news/1141508.

[4] Tom Fawcett. In-vivo spam filtering: A challenge problem for data mining. (section 2.4). http://www.hpl.hp.com/personal/Tom_Fawcett/papers/spam-KDDexp.pdf.

[5] Exim MTA, using DNS Block Lists - http://www.exim.org/howto/rbl.html.

[6] Jeremy Howard. Why the spamcop blocking list is harmful, 2003. Available from http://jhoward.fastmail.fm/spamcop.html.

[7] Mail Abuse Prevention System (MAPS), official site - www.mail-abuse.com.

[8] Adalberto Zamudio. What it is, how it can affect us, and how to deal with spam., 2003. http://www.giac.org/practical/GSEC/Adalberto_Zamudio_GSEC.pdf.

[9] Roland Piquepaille. Why blacklisting spammers is a bad idea, 2003. Available at http://radio.weblogs.com/0105910/categories/sidebars/2003/11/09.html.

[10] David Madigan Rutgers. Statistics and the war on spam. *Statistics, A Guide to the Unknown*, 2003.

[11] David D. Lewis and Marc Ringuette. A comparison of two learning algorithms for text categorization. In *Proceedings of SDAIR-94, 3rd Annual Symposium on Document Analysis and Information Retrieval*, pages 81–93, Las Vegas, US, 1994.

[12] Mozilla, 2004. More information of the Mozilla Thunderbird email client is available from http://www.mozilla.org/products/thunderbird/.

[13] SpamAssasin, 2004. SpamAssasin is an extensible server side email filter. More information available from http://spamassassin.apache.org.

[14] Spamhole, The Fake Open SMTP Relay - http://www.spamhole.net.

[15] Vipul Ved Prakash, 2004. Vipul's Razor is a distributed, collaborative, spam detection and filtering network - http://razor.sourceforge.net.

[16] European Union, 2002. The Full text of the European Union Directive on Privacy and Electronic Communications is available at http://europa.eu.int/eur-lex/en/index.html.

[17] inSourced, 2004. EU states slated for not enforcing anti-spam laws - http://www.in-sourced.com/article/articleview/1565/1/1/.

[18] CAN-SPAM, 2003. Full text of the CAN-SPAM act available from the US Library of Congress at http://thomas.loc.gov/.

[19] Tim McCollum, 2004. USA Tries to Can Spam - Article available at http://www.theiia.org/itaudit/index.cfm?fuseaction=forum&fid=5486.

[20] Amit Asaravala. With this law, you can spam. *Wired News*, 2003. Available from http://www.wired.com/news/business/0,1367,62020,00.html?tw=wn_story_related.

[21] Memorandum of understanding on mutual enforcement assistance in commercial email matters - http://www.ftc.gov/os/2004/07/040630spammoutext.pdf.

[22] ITU Activities on Countering Spam - http://www.itu.int/osg/spu/spam/index.phtml.

[23] Spamhaus. Virus and ddos attacks on spamhaus., 2003. A catalogue of varied attacks on the Spamhaus system - http://www.spamhaus.org/cyberattacks/.

[24] John Leyden. Sobig linked to ddos attacks on anti-spam sites. *The Register*, 2003. Monkeys.com, Compu.net and the SPEWS blocklist closed because of dDoS attacks launched by the SoBig worm - http://www.theregister.co.uk/2003/09/25/sobig_linked_to_ddos_attacks/.

[25] Kyle Cassidy and A. Michael Berman. Can you trust your email? In *Proceedings of the Eastern Small College Computing Conference*, New Rochelle, NY, US, 1995.

[26] Sandvine. Trend analysis: Spam trojans and their impact on broadband service providers, 2004. Report available from http://www.sandvine.com/.

[27] LURHQ, 2003. The Reverse-Proxy Spam Trojan, Migmaf, is described in detail at http://www.lurhq.com/migmaf.html.

[28] A. Hussain, J. Heidemann, and C. Papadopoulos. A framework for classifying denial of service attacks. In *Proceedings of ACM SIGCOMM 2003*, Karlsruhe, Germany, 2003.

[29] Ian Clarke, Oskar Sandberg, Brandon Wiley, and Theodore W. Hong. Freenet: A distributed anonymous information storage and retrieval system. *Lecture Notes in Computer Science*, 2009:46, 2001.

[30] Riccardo Lancellotti, Bruno Ciciani, and Michele Colajanni. A scalable architecture for cooperative web caching, 2002. http://weblab.ing.unimo.it/papers/networking2002.pdf.

[31] Mindi McDowell. Understanding denial-of-service attacks, 2004. Available from http://www.uscert.gov/cas/tips/ST04-015.html.

[32] Distributed.net was one of the earliest projects to harness the combined computing power of 1000s of nodes across the Internet in order to solve various complex problems - www.distributed.net.