University of Tasmania Open Access Repository

Cover sheet

Title

Catching Spam Before it Arrives: Domain Specific Dynamic Blacklists

Author

Cook, DF, Hartnett, JS, Manderson, K, Joel Scanlan

Catching Spam Before it Arrives: Domain Specific Dynamic Blacklists

Duncan Cook, Jacky Hartnett, Kevin Manderson and Joel Scanlan

School of Computing
University of Tasmania
Private Bag 100, Hobart 7001, Tasmania

{cookdf, J.Hartnett, jdscanla}@utas.edu.au, kevin.m@webos.com.au

Abstract

The arrival of any piece of unsolicited and unwanted email (spam) into a user's email inbox is a problem. It results in real costs to organisations and possibly an increasing reluctance to use email by some users. Currently most spam prevention techniques rely on methods that examine the whole email message at the mail server. This paper describes research that aims to deny spam entry into the internal network in the first place.

Examination of live amalgamated audit logs from a Linux kernel firewall, the PortSentry intrusion detection system and the Sendmail mail transfer agents has shown that it is possible that automated mailing programs send characteristic probes to the network gateway just before launching an avalanche of mail. Similarly it seems possible to detect precursor activity from some potential zombie machines. A real time system that could detect such activity needs to be certain that a particular IP address is about to send spam before blocking all of its packets at the network gateway. The architecture for a system that establishes certainty that a particular IP address is about to or has started sending spam is described in this paper. The eventual aim is to recognise precursor activity from spammers in real time, establish certainty that this IP address is about to send or is currently sending spam packets and to then deny packets from this IP address at a range of communicating gateways

1 Introduction

Unsolicited bulk email, commonly referred to as spam, is a major concern for the email infrastructure. Email is now a significant communications channel and could arguably be described as an essential form of communication in today's connected society. As it is now so heavily relied upon, anything negatively affecting the functionality of email severely threatens its usefulness as a communications medium.

It is hard to determine the exact proportion of spam compared with regular email, with various reports listing figures of anywhere between 2.5%-10% (Cranor & LaMacchia 1998) and 60% (MessageLabs 2005). While spam message proportion is difficult to determine, there is no doubt that it is increasing, as the above figures show. Regardless of the specific percentage of email that spam accounts for, it is undoubtedly significant enough that large amounts of time and money are currently being spent to combat the rising tide of spam messages.

This paper will begin by outlining the impact spam has on individuals, companies and the general perception of the usefulness of the email system. Next, the motivation for sending spam will be discussed, along with the techniques currently used by spammers. A new method of spam detection and prevention is then proposed, designed to reduce the amount of resources consumed by spam within a network. The paper concludes with a summary combined with suggestions for future research within the area of network-based spam detection and prevention.

1.1 Impact of Spam

Spam wastes time, money and resources. Manually removing spam messages from an inbox wastes a user's time and if that user is at work then it is also costing their employer money in lost productivity. Conservative estimates indicate that the total cost of spam on users (worldwide) in 2001 was €10 billion (approximately AU$16 billion) a year (Gauthronet & Drouard 2001). Other reports estimate that spam cost US companies alone $10 billion (approximately AU$13 billion) in lost productivity in 2003 (Bekker 2003).

Spam also consumes valuable computing resources. Every unsolicited email consumes bandwidth and network resources regardless of whether users actually receive it (Whitworth & Whitworth 2004). Spam messages cause delays for all Internet users as they waste resources on the Internet backbone. Furthermore, as some spammers use dictionary attacks and outdated address lists, many messages are rejected as being invalid by the receiving mail server and "bounced" back to the sender (Garcia et al. 2004), wasting even more Internet resources. Also, running any sort of spam filter on a mail server steals processing time from the server's major purpose: delivering email. It is worth noticing that the people who are making money out of spam are generally not the people who bear the full impact of dealing with the spam messages.

A further cost of spam is its psychological effect on users regarding their willingness to communicate via email. According to a report published by the PEW Internet & American Life Project, 25% of email users surveyed admit that the ever-increasing volume of spam has reduced their overall use of email with 60% of those saying that it has reduced their email use in a big way (Fallows 2003). Furthermore, 30% of email users are concerned that their filtering systems may block incoming legitimate messages and 23% are concerned that their emails to others may be blocked by filtering systems. These concerns initially seem rather conservative or even paranoid, but the truth is they are inevitably grounded in reality; customers of various Australian Internet service providers (ISPs) recently ended up having legitimate email blocked by Telstra's BigPond mail servers after their ISPs were blacklisted by Telstra's spam filtering system (LeMay 2005). The numbers above also serve to further highlight the effect spam has on the public's perception of the usefulness of email. If the amount of spam saturating users' inboxes continues to increase, there is a very real risk that the general public will eventually abandon email for other, less frustrating communication mediums.

1.2 Spammer Motivation and Techniques

When the above impacts and costs of spam are taken into account, it is feasible to ask the question: *'Why does spam continue to exist?'* The answer to this is simply that no matter how strange a concept it may seem, spam gets results. A small percentage of email users actually do buy products advertised through spam emails. While the public perception of spam is largely negative, spammers would not be operating if it were not a viable source of income. Weiss (2003) notes that a spammer only needs to receive one hundred responses out of ten million spam messages (0.001% acceptance) to turn a profit. As the acceptance of offers contained in spam emails is generally proportional to the amount of people that the email is sent to, it makes economic sense for spammers to send the message to as many people as possible.

In order to reach a large volume of users, spammers require an equally large number of email addresses. These are usually collected in three different ways: by using programs known as spam-bots to scavenge for email addresses listed on web sites and message boards (particularly USENET groups), by performing a dictionary attack (pairing randomly generated usernames with known domain names to 'guess' a correct address) or by purchasing address lists from other individuals or organisations (Pfleeger, S. L. & Bloom 2005).

Once they have addresses, spammers can use programs known as "bulk mailers" to automate the sending of spam. These programs can send huge volumes of email messages in a small amount of time. Some bulk mailing programs use open-relays (email servers that allow unauthorised users to send email) to send messages, effectively hiding the true address of the spammer. Bulk mailers can also fabricate the *from* address in email message headers to further hide the identity of the spammer (Garcia et al. 2004).

Another technique spammers utilise to send emails is with the use of *zombie networks*, also know as *bot networks*. Zombie is the term given to a computer that has been infected by a virus, worm, or Trojan Horse (Levy 2003), which allows remote entities to take control and use it for their own (usually illegal) purposes. A large amount of these computers, usually called a *network* or *army* can be co-opted to send spam emails, requiring little of the spammer's own computing power and network bandwidth. This technique is also popular as it protects the identity of the spammer (Paulson 2004).

2 Current Spam Detection and Prevention

Techniques currently employed to prevent the arrival of spam generally revolve around the use of filters. Filters examine various parts of an email message to determine whether or not it is spam. Filtering systems can be further classified based on the parts of the email messages they use for spam detection. Origin or address-based filters typically use network information for spam classification, while content filters examine the actual contents of email messages.

2.1 Origin-Based Filters

As mentioned above, origin-based filters use network information in order to detect spam. IP and email address are the most common pieces of network information used. The three major types of origin-based filters are blacklists, whitelists and challenge/response systems. Other spam prevention techniques that broadly fall into the category of origin-based filters are sender authentication systems such as the Sender ID Framework (Lyon & Wong 2004) and DomainKeys (Delany 2005). These systems place more emphasis on the authentication of the sender or the sender's domain but require modifications to the existing email system to be effective.

2.1.1 Blacklists

Blacklists, also known as *realtime blackhole lists* (RBL) or *domain name system black lists* (DNSBL), can filter mail from mail servers or domains that have sent spam or are suspected of doing so. IP addresses of known or suspected spammers are entered into centrally maintained databases and made available as blacklists through the Internet. Standard DNS lookups are used to query these databases at the time of SMTP connection or when mail is received, with spam classification occurring based on the reply given.

Blacklists are managed by various separate groups; each with its own focus and different policies in regards to how an IP address gets on (and off) the list (Allman 2003). Blacklist focus could include RFC compliance, open relays, open proxies (proxy servers that allow anyone to connect to them and send spam), IP addresses or domains that spam has actually come from or even dial-up users (as most legitimate mail servers are not run over dial-up connections). Blacklists also differ in how aggressive they are at categorizing spam sources; some lists prioritise avoiding false positives (legitimate email classified as spam), while other more aggressive

blacklists aim to catch the largest percentage of spam (which usually produces more false positives). Popular blacklists include Trend Micro's RBL+ (formally MAPS) (*Trend Micro RBL+ Service* 2005), the SPEWS list (*Spam Prevention Early Warning System* 2005) and the Spamhaus SBL and XBL lists (*The Spamhaus Project* 2005).

As blacklists only require DNS lookups, they have a very low CPU overhead and are generally easy to implement. Another advantage of blacklists is that they allow spam to be blocked at the SMTP connection phase, effectively stopping it from entering the network. Blacklists are not without disadvantages however, an example of such is the fact that they are maintained by an external entity. These lists could potentially be removed at any time without warning, leaving networks solely relying on these blacklists without any form of spam protection at all. The effectiveness of a blacklist relies on the people who manage them; if blacklists are not updated in a timely manner, spam can get through. Additionally, some blacklist providers (particularly SPEWS) neglect to specify the policies used to add and remove addresses from the list, effectively forcing network administrators who use these lists to trust the judgement of other people.

Another problem with blacklists is that as the amount of spam increases, the number of DNS lookups to check blacklists increases. This is particularly bad for mail servers that use more than one blacklist. A study conducted at the MIT Computer Science and Artificial Intelligence Laboratory found that blacklist DNS lookups accounted for 14% of all their DNS lookups in 2004, up from less than 0.4% in 2000 (Jung & Sit 2004).

Spammers can circumvent blacklists to a certain degree by using *zombie networks* (as described in Section 1.2 above). As a *zombie network* comprises of many different computers, all of which could be from different domains, a blacklist on a specific domain would provide only minimal spam protection.

2.1.2   Whitelists

Whitelists allow users to specifically define "trusted" addresses that will immediately classify as legitimate all email received from those addresses. An appealing quality of whitelists is that for most users a whitelist would be significantly smaller and easier to maintain than a blacklist. Also, mail flagged by a whitelist as legitimate can bypass further spam filters, effectively reducing the load on those filters.

A problem with whitelists, however, is that since the sender of email messages is not authenticated, spammers who can guess an address on the whitelist can then freely propagate spam to that address (Allman 2003). Additionally, if used by themselves, whitelists can tend to be overly restrictive as it is almost inevitable that legitimate mail will eventually be blocked or filtered into a lower priority mailbox. If this lower priority mailbox contains a large amount of spam, searching for valid messages could become a very difficult task. Whitelists are, therefore, best used when combined with other spam blocking techniques (Garcia et al. 2004).

2.1.3   Challenge/Response Systems

Challenge/response systems are an advanced version of whitelists, allowing senders who are not on the whitelist to have their emails received. Incoming messages from addresses not on the whitelist trigger an automatic reply (or *challenge*) to the sender, requiring them to prove that they are a real user and not an automated mailer. For example, the sender may be required to click on a link in the reply message and enter a valid email address and the ID number of the response message. If this process is completed, then the email successfully passes through the challenge/response system (Pfleeger, S. L. & Bloom 2005).

The challenge/response method aims to protect against automated mailer programs by forcing the user to complete a task that is simple for a human but too complicated for a program to handle. Challenge/response systems also protect against spammers who manually send email, as the time required to complete the challenge could be better used sending spam to additional addresses. Challenge/response systems also help to protect against the generally large amount of false positives generated by traditional whitelist systems.

One problem with whitelists is the issue of deadlock. If two parties who have never corresponded before both run challenge/response systems, the challenge sent by the recipient's system will be caught by the sender's challenge/response system and neither party will have the opportunity to provide an appropriate response (Barracuda Networks (Date Unknown)). This problem could be alleviated if the original sender adds the recipient's address to their whitelist before commencing communication.

Another problem associated with the use of challenge/response systems is legitimate automated email lists that the user has subscribed to. These lists cannot respond to the challenge messages generated by the system, and mail from these sources may be marked as spam. As with the deadlock issue, this problem could be alleviated if the subscriber adds the mailing list address to their whitelist before subscribing to the list.

2.2   Content Filters

While origin-based filters such as whitelists and blacklists examine email headers and other network information, content filters detect spam by looking inside the email and examining the message contents. Most content-based spam detection systems try to "understand" the text to various extents in order to identify spam (Allman 2003). A simple word filter, for example, could look to see if the message contains the words *Viagra* or *sex* or the phrases "*buy now*" or "*you've won*" to determine whether it is spam. Filters based on this technique are commonly called keyword-based filters. These filters can be highly context sensitive though, as a pharmacy may not want emails with the word *Viagra* in them filtered out. Popular content filter types include Bayesian filters and rule-based filters.

2.2.1 Bayesian Filters

Spam filtering systems using Naïve Bayesian classification were originally proposed by two separate parties at the AAAI-98 Workshop on Learning for Text Categorization; one was by Pantel and Lin (1998) and the other was by Microsoft Research (Sahami et al. 1998). Bayesian (also known as statistical) filters work by analysing the words inside an email message to calculate the probability that it is spam. This probability is based on not only those words that provide evidence that a message is spam, but also on those words that provide evidence that a message is not spam. Words that are not generally found in spam messages contribute to the probability value in very much the same way as words that are frequently found in spam messages (Graham 2003).

To calculate an email's spam probability with a good degree of accuracy, Bayesian filters need to be "trained" by being given examples of what constitutes a spam email and what does not. The advantage of this technique is that, given appropriate time and training data, Bayesian filters can achieve a combination of extremely high accuracy rates with a low percentage of false positives (Graham 2003). The low amount of false positives generated by a Bayesian filter is useful, as users generally regard the classification of legitimate emails as spam as an order of magnitude worse than receiving spam incorrectly classified as legitimate (see Section 1.1).

A further advantage of Bayesian filters is that they are constantly self-adapting. Provided they receive ongoing training data from the user, Bayesian filters evolve to stop new spam techniques.

The problem with Bayesian filters is that, like all other content filters, they require the entire message to be received before analysis can begin. Furthermore, Bayesian filters are generally more resource intensive than origin-based filters, as calculating Bayesian probabilities requires significantly more processing power than simply querying a list.

2.2.2 Rule-Based Filters

Rule-based, or heuristic, filters search the email message for patterns that indicate spam. These patterns could include specific words or phrases, malformed message headers and large amounts of exclamation marks and capital letters. Detection of a specific pattern attributes an amount of points to an email message and once the point value of an email exceeds a set threshold, it is classified as spam. Rule-based filters were the most common type of spam filter until 2002, when Bayesian filters became popular (Graham 2003).

The problem with rule-based filters is that, since the rule set is largely static, they are easily defeated by spammer techniques such as word obfuscation. For example, a rule-based filter that checks for the word *Free* will not be able to detect the string *F*r*e*e* as spam. It is incredibly difficult to include rules for every possible misspelling of common spam words, which has limited the effectiveness of such filters.

2.3 Other Filters

While the above filters are the most commonly used spam filtering techniques, it is by no means an exhaustive list. Other data mining, machine learning and text classification techniques currently under research include: digest-based filters (Damiani et al. 2004), density-based filters (Yoshida et al. 2004), Chi-squared filters (O'Brien & Vogel 2003), global collaboration filters (Hulten et al. 2004) and artificial neural networks (Drewes 2002). Social network techniques, such as Reputation Network Analysis are also under investigation (Golbeck & Hendler 2004).

It should also be noted that the various spam filtering methods described above are by no means mutually exclusive. A popular spam filtering program, SpamAssassin (*The Apache SpamAssassin Project* 2005), uses a combination of Bayesian filtering, rule-based filtering and blacklist checking to calculate a spam score for a particular email message. Messages that exceed a particular user-defined threshold are then marked as spam and dealt with appropriately.

3 Proposed Work

The new spam protection technique proposed in this paper revolves around using intrusion detection system (IDS) techniques, specifically audit log analysis, to stop spam messages before they enter the network. Network precursors such as portscans are used as evidence that a particular IP address will soon send spam. Once enough evidence against a particular IP address is gathered, rules will be added to the network firewall to block SMTP connections from that address. We term this system a domain specific dynamic blacklist (DSDBL), as it essentially uses network information to dynamically blacklist the IP addresses of spammers attacking our domain.

An interesting aspect of this technique is that it aims to stop spam at the network gateway. The filtering technologies described above can be deployed at various parts of the network, including at a company's ISP, their local mail server or at each user's computer, but there has been little research conducted into stopping spam at the gateway level. Most filtering techniques (with the notable exception of blacklists) wait until the entire message has been received before taking action. This places a great burden on a company's mail servers, as every mail, spam or legitimate must be received and stored before filters remove the spam. It would be more efficient to deny these spam messages entry into the network in the first place. This would result in reduced mail server load, increasing the operational efficiency of an organisation's email system.

Another advantage of gateway-based protection is that data from all of the network's gateways can be correlated, potentially identifying attacks that occur over multiple gateways that service a particular IP space (Scanlan et al. 2004). Providing protection at the gateway level further contributes to the idea of a *layered defense* (Pfleeger, C. P. & Pfleeger 2003), with spam potentially having to evade filtering at the ISP level, the gateway level, the

mail server level and at the user's machine to get into a user's inbox.

3.1 Audit Logs

In order to create a system that can detect and respond to spam attacks in real-time, there needs to be a facility for the gateways and mail servers to record what is happening at any given time. This facility is provided by the use of *audit logs*. Audit logs have long been used in computer security to detect intrusions; Clifford Stoll, for example, used system printouts to detect and track a hacker intruding into the Berkeley University computer network (1991).

Audit log information can come from a variety of sources, including operating systems, firewalls, routers, mail servers and third party software (Amoroso 1999). As all these sources can produce a large amount of information, choosing the right system events to audit is of crucial importance. There is always a trade-off between the amount of information a system collects (how thorough the audit log will be) and the amount of system overhead the logging processes and stored log files use (Kemmerer & Vigna 2002). Generating too much log data makes it harder to analyse all the information, while not generating enough data may lead to an ineffective system. The primary sources of audit log data used in this research project are syslog files, generated by the Linux kernel firewall, the PortSentry intrusion detection system and the qmail and Sendmail mail transfer agents.

The UNIX system logger (*syslog*) is a commonly used logging application and it provides the log sources for this research. *Syslog* provides a way for different processes, applications and devices to send log information to a centralised point, known as the *syslog* server (Lonvick 2001). The three distinct sources that *syslog* events come from are: processes running on the local machine (the machine running the *syslog* daemon), kernel routines running on the local machine and processes running on other machines. All messages contain the source of the message, the authorizations associated with the message, the priority assigned to the message and the content of the message. For every message sent to the *syslog* server, a timestamp and message type keyword is appended, along with a new line character at the end. The syslog.conf file is consulted and the message is then handled in one of the following ways: sent to a file or specific UNIX device, sent to a user (e.g. root) or all users, sent to a program using the UNIX *pipe* command or sent to another machine (Amoroso 1999).

The audit logs that have been provided for this research project come from a network covering an almost complete class C IP domain. This network is similar in size to a typical small hosting provider or business in Australia. Internally, the network comprises of multiple gateways connected to multiple mail servers.

3.2 Phase 1

The first system was developed to perform initial audit log analysis and to identify what sort of network activity,

if any, occurred as a precursor to spam messages. The system was developed in the *Perl* scripting language, using *Perl's* powerful regular expression engine to traverse an amalgamated audit log spanning multiple gateways and mail servers.

Phase 1 was primarily needed because of our use of "live" audit log data. As the audit log files contain an extremely large amount of text (for example, a 21 day log file used in this research contained approximately 300 MB of text), they cannot be sorted through manually. Therefore, a system was needed to reduce the logs to a manageable size and in the process, identify the sort of network activity to examine in Phase 2. **Figure 3.1** (below) shows an overview of the system.
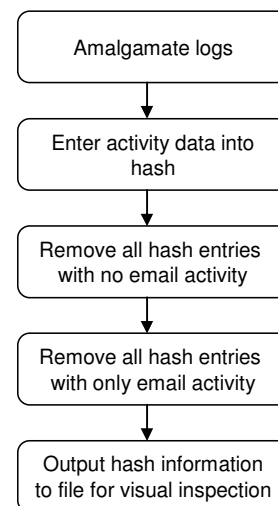

Figure 3.1: Phase 1

The first stage was to amalgamate log data from all gateways and mail servers into a single file. Entries were ordered by date and time, simulating the output that would result if all gateways and mail servers wrote log data to the same file.

The second stage was to traverse the log and place entries into a *Perl* hash table data structure (essentially an associative array), indexed by IP address. Within each IP hash table entry, all log lines relevant to that IP address were stored in an array.

The first stage of data reduction was to remove all IP entries from the hash table that contained no email activity. No email activity meant that the IP has not sent any spam (in the time period covered by the log file) and was therefore not relevant to our results.

The second stage of data reduction was to remove all IP entries that only contained email activity. The logic behind this was that we are only interested in IP addresses that have other precursor network activity entries.

The final stage of the Phase 1 program was to output the remaining hash entries into an output text document for visual inspection.

A hash table was used as it offered increased searching speed in data entry and extraction stages at the expense of high memory utilisation. Memory utilisation was not of high concern as this Phase 1 system was never intended

to run in real time. The computer used in this stage contained a Pentium III 800MHz CPU with 512 MB of RAM. A 21 day audit log 303 MB in size was processed in approximately 10 minutes.

3.2.1 Phase 1 Results

The Phase 1 system was used to process two audit log files; the first log (Log 1) representing a 21 day period between July 1 and July 21 2004 with the second log (Log 2) representing a 31 day period between August 1 and August 31 2004.

The major issue we were presented with when recording results from Phase 1 testing was that of spam classification. The sendmail entries in the log file contained only a small amount of information about the received mail message. We eventually decided to use three fields of the sendmail entry to determine spam; *from*, *msgid*, and *relay*. Based on the information listed in each of these fields, a decision was made as to whether a particular message was spam. For example, if a message had no *from* address, it most likely was spam, as legitimate email programs generally include this information when an email is sent. Also, the domain information included in the *from* address (if it contains any data at all) was checked against the domain listed in the *msgid* and *relay* fields; inconsistent domain information can be an indicator that an email is spam, although it must be noted that this is not always the case. In order to take this into account, *msgid* and *relay* information was only examined if the *from* address looked suspicious or contained no information.

In Log 1, 123 spam messages from 98 distinct IP addresses were recorded as having precursor activity. 6.98 precursors were received per spam message on average. The precursor activity recorded consisted of portscans across a variety of TCP ports. The major ports that were scanned as precursors to the sending of spam were 135, 139, 1433, 25 and 0. Log 2 recorded 223 distinct IP addresses sending a total of 312 spam messages with precursors. On average, 10.29 precursors appeared per spam message, with the majority of activity recorded on TCP ports 135, 139, 1433 and 0. This clearly shows that precursor activity for email messages was observed, although the proportion of messages with recorded with precursor activity was extremely small when compared to the overall count of mail in the time period covered by the audit log. Only 0.09% of emails from Log 1 contained any network precursor activity at all and Log 2, although offering a significant precursor increase over Log 1, still only reported precursors for 0.28% of all emails.

TCP port 135 is officially used for Microsoft Remote Procedure Call (also known as Distributed COM Service Control Manager), while port 139 is officially used for the NetBIOS Session Service (Seifried 2003). These services are both used for SMB file and print sharing, but more recently TCP ports 153 and 139 have received attention as ports exploited by the W32/Blaster worm (Dougherty et al. 2003). Port 1433 is reserved for use by the Microsoft SQL server but is also targeted for exploitation by another Internet worm known as Spida, SQLSnake or Digispid (Dougherty & Householder 2002).

The interesting precursors appear on TCP ports 25 and 0. Port 25 is used for SMTP connections (i.e. the transfer of email messages) so activity on this port could represent a remote entity scanning to see if a mail server is operating. Port 0 is listed as a reserved port by IANA, meaning that no activity should appear on this port. Even though it is reserved, some systems treat a port 0 connection attempt as a request for connection on the lowest free TCP port. Additionally, due to the differing responses generated by various operating systems when a port 0 connection is requested, port 0 scanning can also be used to determine the OS running on a target machine (Jones 2003). This observed activity on port 0 could represent an automated mailer testing to see if a computer is located at the target IP address, then sending spam to that address if the target is confirmed.

Another observed pattern was that for spam that appeared after precursor attacks consistent with worm activity (ports 135, 139 and 1433), the message usually arrived a number of days after the last precursor activity. Conversely, spam that appeared after port 0 and 25 activity usually arrived within a number of seconds.

It could be inferred that precursor activity on the commonly exploited ports is most likely indicative of zombie machines compromised by the worms listed earlier. The initial (precursor) activity from these hosts most likely represents the worm attempting to replicate, while the subsequent spam messages result from the compromised machines being mobilised to propagate spam. As the precursor activity observed on ports 0 and 25 occurred much closer to the actual arrival of spam messages, this activity could indicate the use of a bulk mailer program that scans for active IP addresses to propagate spam to.

Despite these findings, the fact that precursor activity was only observed in a small number of all emails indicates that detecting spam based on precursor activity alone will most likely offer only a minute improvement on current systems, at a significant increase in computing power needed to run the system. For the real-time system to be viable, an alternate method of determining when spam is likely to arrive needs to be developed.

3.3 Phase 2

The Phase 1 system was able to analyse audit logs from multiple gateways and mail servers to determine if and what precursors appeared before spam messages arrived. This system did not operate in real-time, however, as it had to index an entire log file into the hash table before potential precursor activity could be identified through manual inspection of the program's output. The goal for Phase 2 is to construct a system that can operate in real-time, dynamically detecting new spam precursors and taking appropriate action against IP addresses that are identified as sending spam. Once a precursor pattern is determined, we want the system to identify the occurrence of precursor activity and block all email from the offending IP address from entering the network. This

will allow the system the flexibility to adapt to the ever changing landscape of network precursors to spam.

Due to the low percentage of network activity observed as spam precursors in the results from Phase 1 (see Section 3.2.1), an alternative method was sought to determine if spam is likely to be received by the network. The solution implemented in the Phase 2 system was to use spam messages themselves as an indicator that more spam is likely to arrive from an IP. This allows the system to identify IP addresses that are sending multiple spam messages to different addresses within the network or sending multiple spam messages to the same address. A criticism of this approach could be that it requires the network to receive multiple spam messages before action is taken. While this is true and the obvious solution could be to just ban a spammer's IP address as soon as the first spam is flagged by the mail server's spam filter, it is believed that this method provides the end user with greater protection from false positives. As spam filtering at the network wide level generally involves no user input at all, it is important to do as much as possible to prevent legitimate email from being marked as spam and blocked from the network.

The Phase 2 system will build upon techniques developed for the Phase 1 system, making necessary modifications for it to run in real-time. One problem with Phase 1 which prevented fully automated operation was the identification of spam. As noted earlier, email *from*, *msgid*, and *relay* fields as reported in the mail log were examined as the primary source for identification information. For a real-time automated system, more evidence is required to determine whether an email is spam before precursor detection can begin. To this end, it was decided to use the spam filter deployed on the mail server (in our case it is SpamAssassin) to provide the initial spam identification. This allows for a much greater degree of certainty that a particular email is spam than could have been gained through the limited information contained in the mail server log.

Another necessary modification of the Phase 1 system was to move away from the large hash to store log information. The hash table created in the Phase 1 system consumed far too much memory to be used on operating gateways. It was decided that a better solution for a real-time system is to store this information in a database. While the time required to insert and extract data from a database is significantly greater than using a hash table set up in memory, the use of a database offers greater robustness and flexibility. This robustness is provided by the fact that a database stores information on secondary storage, whereas the hash was contained solely in memory. If the computer running the program crashed or had to be rebooted for another reason, all the data stored in the hash table would be lost and the program would have to be restarted with no knowledge of previous events. Conversely, the use of a database would allow the program to start from where it left off after the computer was restarted as all of the data stored would still exist on secondary storage.

The use of a database also allows the system to be more flexible, as many different processes can access the data stored within the database. This will provide support for separate programs to take action based on data in the database and also perform cleaning functions to prevent the database tables and firewall rules from using too many resources.

The last major modification required of the Phase 1 system involves taking action against those IP addresses that are sending spam. As the Phase 1 system was intended to identify the existence and attributes of spam precursors, no consideration was made in regards to what the system will do once a precursor has been identified. Mail servers, especially those of large ISPs could very possibly be sending a mixture of both spam and legitimate email. A mechanism is needed to ensure that action is only taken against IP addresses that send a large proportion of spam or those that do not send legitimate email. To this end, we propose the use of the method of Sequential Hypothesis testing (Wald 1947), as used by Jung et al. (2004) with the development of their Threshold Random Walk algorithm. Essentially, Sequential Hypothesis Testing is a method for defining two hypotheses (a simple hypothesis and an alternative) and testing via successive observations to determine whether either hypothesis has been reached. In the case of our system, we select between two hypotheses, namely that a given IP address is either malicious or benign. This is accomplished by calculating and assigning suspicion values to IP addresses. Only once an IP address has been classified as malicious (i.e. exceeded a particular spam suspicion threshold) will action be taken to block traffic from that IP into the network. This should provide the system with the scope to allow for large ISPs with a small amount of spammers to not have all of their legitimate email blocked. This should at least minimise or perhaps eliminate the possibility that legitimate emails will be blocked by the system. This is an important issue as the blocking of legitimate emails affects user's willingness to use the technology, as mentioned above in Section 1.1.

3.3.1   System Description

In its default state, the system has to wait for the spam filter to classify a received email as spam. This is accomplished through the use of *Perl*'s *Tail* module (Grabnar 2004). *Tail* allows the system to monitor the spam filter log file and process each entry as it is added to the log. When a new log line appears, the system uses a *Perl* regular expression to extract the timestamp and IP address related to the spam message. The spam filter is used as the primary method of identifying spam emails, effectively eliminating Phase 1's spam identification problem (see Section 3.2.1). Assuming that the spam filter is appropriately effective, this provides a more accurate assessment of whether a particular email is spam.

The system then searches the gateway log up until the time of the spam filter log entry and adds all network activity from the spam IP address into the database. This includes updating the sequential hypothesis information for the classification of the IP address. If the system already has precursor activity recorded for the IP address, the search through the gateway log will start at the line

after the line relating to the last recorded precursor activity. This prevents the system from recording the same precursor activity multiple times per unique IP.

If the IP address has exceeded either of the two threshold values, it is classified malicious or benign, depending on which threshold was exceeded. If the IP address has been classified as benign, it will be added to a "Benign IP" table in the database. This table will record IP addresses that we do not want to block from the network. If an email is received from an IP address listed in the Benign IP table, it will be let through and not scrutinised any further. Precursor activity for IP addresses in the Benign IP table will not be recorded until the IP address is removed from the table.

If the IP address has been classified as malicious, this information will be communicated to all gateways in the network. Rules will be added to each gateway's firewall to deny traffic from the malicious address entry to the network. A diagram of the entire process is shown in **Figure 3.2** below:
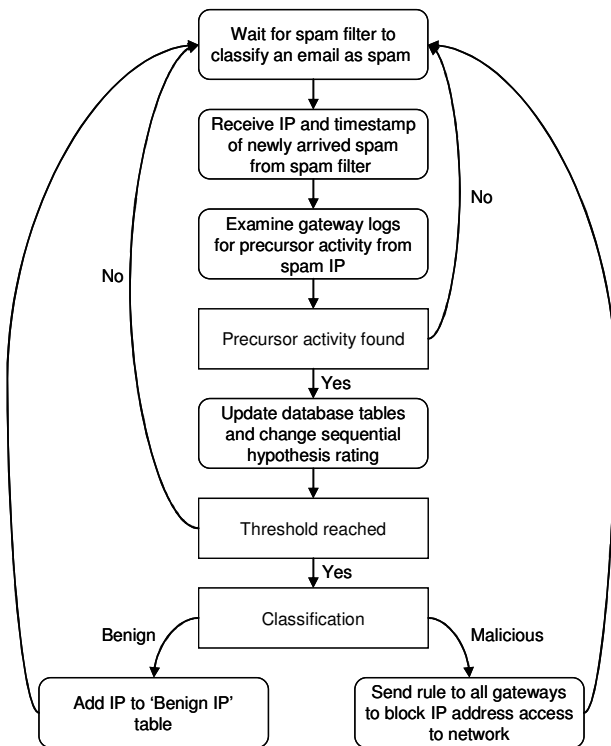

Figure 3.2: Phase 2

As the system is intended to run in real-time, ensuring system efficiency is essential. Every extra database entry adds a small amount of seek time for every database query. Also, every firewall rule added at the gateways increases the amount of time needed to filter every packet attempting to access the network. Therefore, the key to keep the system running efficiently is to remove database entries and firewall rules that are no longer needed. When an IP address is either placed in the Benign IP table or banned from the network, a calculation needs to be made to determine the length of time the IP will stay at either classification. This will allow firewall rules to be removed once an IP address has been banned from the network for an appropriate amount of time. It also allows

for the possibility of a Benign IP becoming more malicious over time.

3.3.2 Expected Phase 2 Results

The Phase 2 system is designed to test whether precursor detection at the gateway is a viable spam protection technique. To answer this, two areas need to be examined: efficiency and timeliness. It needs to be determined if the system will block enough spam messages to warrant the drain it places on gateway resources. If the system only manages to block a small proportion of spam messages, it would make more sense to let the mail server filter every message (as it currently does) and lower the load on the gateways. A further test of viability is the determination if the system can respond to attacks fast enough to stop subsequent spam. If the system cannot identify and respond to precursors in a timely manner, the spam predicted by the precursor may get through before measures are taken to block it from the network.

Another aspect of the system that needs to be determined in order for it to operate efficiently is gateway log examination time. This refers to the amount of time that the system will traverse the gateway audit log looking for precursors. The value for this impacts both the accuracy of the system and its timeliness. Not looking back through the gateway log far enough increases the potential for precursor activity to be missed, while looking too far back will increase the amount of time the system takes to identify precursors.

A further consideration for the system is how long IP addresses are blocked from the network. Blocking an IP for too long increases the possibility that legitimate traffic from that IP may be stopped. Blocking an IP address for an excessive amount of time could also impact on the performance of the network, as the amount of rules the firewall has to check against affects its speed of operation. These considerations can also be applied to the length of time an IP will spend on the "Benign IP" list. We intend to examine the techniques used by Scanlan et al. (2005) to determine if a similar system can be adapted for our use.

4   Discussion and Future Work

The completed system aims to be able to reduce the load on mail server content filters by stopping spam before it enters the network. It is hypothesised that the real time detection of precursor activity combined with initial knowledge from mail server content filters will be able to reach a certain determination that a particular IP address is currently sending spam in sufficient time to stop the bulk of the spam packets from entering the network.

Preliminary work has shown that for some spam attacks there are initial probes from the 'about to spam' IP that may be characteristic. The challenge is to build a system that can detect any such precursor activity, confirm that it is a precursor to a spam attack and block the offending IP in time to stop the spam entering the network.

One limitation of the currently proposed system is that at least one spam message is let in to the network per distinct IP address, as this is the current trigger for precursor investigation. It would be desirable for the system to develop a "signature" of precursor activity once an IP address has been blocked. This information could then be used to block other IP addresses that exhibit this signature activity before they have the chance to send spam.

Another limitation of the currently proposed system is that it performs all if its analysis at a single location. If the host running our system is compromised or shut down, the network will no longer be protected. Distributing the system across a number of gateways could make it more robust, as each gateway could do a proportion of the spam correlation. The system could still run (albeit at a reduced capacity) in the event of gateway failure by one or more gateways running the system. A distributed form of the system also has the potential to be more efficient, as the log analysis load could be shared across multiple machines. This could theoretically decrease the time the system takes to react to the arrival of spam.

5   References

Allman, E 2003, 'Spam, Spam, Spam, Spam, Spam, the FTC, and Spam', *Queue*, vol. 1, no. 6, pp. 62-9.

Amoroso, EG 1999, *Intrusion detection : an introduction to Internet surveillance, correlation, traps, trace back, and response*, 1st edn, Intrusion.Net Books, Sparta, N.J.

The Apache SpamAssassin Project, 2005, viewed July 19 2005, <http://spamassassin.apache.org/index.html>.

Barracuda Networks (Date Unknown), *An Overview of Spam Blocking Techniques*, Barracuda Networks, viewed Aug 22 2005, <http://www.barracudanetworks.com/ns/downloads/barracuda_spam_blocking_techniques.pdf>.

Bekker, S 2003, *Spam to Cost U.S. Companies $10 Billion in 2003*, ENT News, viewed May 11 2005, <http://www.entmag.com/news/article.asp?EditorialsID=5651>.

Cranor, LF & LaMacchia, BA 1998, 'Spam!' *Commun. ACM*, vol. 41, no. 8, pp. 74-83.

Damiani, E, Vimercati, SDCd, Paraboschi, S & Samarati, P 2004, 'An Open Digest-based Technique for Spam Detection', paper presented to The 2004 International Workshop on Security in Parallel and Distributed Systems, San Francisco, CA USA.

Delany, M 2005, *Work in Progress, Internet-Draft: Domain-based Email Authentication Using Public-Keys Advertised in the DNS (DomainKeys)*, Internet Engineering Task Force, viewed May 6 2005, <http://www.ietf.org/internet-drafts/draft-delany-domainkeys-base-02.txt>.

Dougherty, C & Householder, A 2002, *CERT® Incident Note IN-2002-04*, CERT Coordination Center, viewed Aug 22 2005, <http://www.cert.org/incident_notes/IN-2002-04.html>.

Dougherty, C, Havrilla, J, Hernan, S & Lindner, M 2003, *CERT® Advisory CA-2003-20 W32/Blaster worm*, CERT Coordination Center, viewed Aug 22 2005, <http://www.cert.org/advisories/CA-2003-20.html>.

Drewes, R 2002, *An artificial neural network spam classifier*, Rich Drewes, viewed May 8 2005, <http://www.interstice.com/drewes/cs676/spam-nn/spam-nn.html>.

Fallows, D 2003, *Spam. How it is Hurting Email and Degrading Life on the Internet*, Pew Internet & American Life Project.

Garcia, FD, Hoepman, J-H & van Nieuwenhuizen, J 2004, 'Spam Filter Analysis', paper presented to 19th IFIP International Information Security Conference, Toulouse, France.

Gauthronet, S & Drouard, É 2001, *Unsolicited Commercial Communications and*

Data Protection, Commission of the European Communities.

Golbeck, J & Hendler, J 2004, 'Reputation Network Analysis for Email Filtering', paper presented to Conference on Email and Anti-Spam (CEAS), Mountain View, CA, USA, July 2004.

Grabnar, M 2004, *File :: Tail - Perl extension for reading from continously updated files*, viewed Aug 24 2005, <http://search.cpan.org/~mgrabnar/File-Tail-0.99.1/Tail.pm>.

Graham, P 2003, 'Better Bayesian Filtering', paper presented to 2003 Spam Conference.

Hulten, G, Goodman, J & Rounthwaite, R 2004, 'Filtering spam e-mail on a global scale', in *Proceedings of the 13th international World Wide Web conference on Alternate track papers \& posters*, ACM Press, New York, NY, USA, pp. 366-7.

Jones, S 2003, *Port 0 OS Fingerprinting*, Network Penetration, viewed Aug 22 2005, <http://www.networkpenetration.com/port0.html>.

Jung, J & Sit, E 2004, 'An empirical study of spam traffic and the use of DNS black lists', in *Proceedings of the 4th ACM SIGCOMM conference on Internet measurement*, ACM Press, Taormina, Sicily, Italy, pp. 370-5.

Jung, J, Paxson, V, Berger, AW & Balakrishnan, H 2004, 'Fast Portscan Detection Using Sequential Hypothesis Testing', paper presented to IEEE Symposium on Security and Privacy, Oakland, California, USA, 9-12 May.

Kemmerer, RA & Vigna, G 2002, 'Intrusion detection: a brief history and overview', *Computer*, vol. 35, no. 4, pp. 27-30.

LeMay, R 2005, *Spam sees Westnet blocked by BigPond*, ZDNet Australia, viewed Aug 15 2005, <http://www.zdnet.com.au/news/communications/soa/Spam_sees_Westnet_blocked_by_BigPond/0,2000061791,39204739,00.htm>.

Levy, E 2003, 'The making of a spam zombie army. Dissecting the Sobig worms', *Security & Privacy Magazine, IEEE*, vol. 1, no. 4, pp. 58-9.

Lonvick, C 2001, *RFC 3164: The BSD Syslog Protocol*, Network Working Group, May 9, <http://www.faqs.org/rfcs/rfc3164.html>.

Lyon, J & Wong, M 2004, *Work in Progress, Internet-Draft: Sender ID: Authenticating E-Mail*, Internet Engineering Task Force, viewed May 6 2005, <http://download.microsoft.com/download/6/c/5/6c530 77f-013e-480c-a19d-787850d84861/senderid_spec1.pdf>.

MessageLabs 2005, *MessageLabs Email Threats - Overview*, MessageLabs, viewed Aug 15 2005, <http://www.messagelabs.co.uk/publishedcontent/publi sh/threat_watch_dotcom_en/threat_statistics/spam_inte rcepts/DA_114633.chp.html>.

O'Brien, C & Vogel, C 2003, 'Spam filters: bayes vs. chi-squared; letters vs. words', in *Proceedings of the 1st international symposium on Information and communication technologies*, Trinity College Dublin, Dublin, Ireland, pp. 291-6.

Pantel, P & Lin, D 1998, 'SpamCop: A Spam Classication & Organization Program', paper presented to AAAI-98 Workshop on Learning for Text Categorization.

Paulson, LD 2004, 'Spam hits instant messaging', *Computer*, vol. 37, no. 4, p. 18.

Pfleeger, CP & Pfleeger, SL 2003, *Security in Computing*, 3rd Int edn, Prentice Hall PTR, Upper Saddle River, N.J.

Pfleeger, SL & Bloom, G 2005, 'Canning Spam: Proposed Solutions to Unwanted Email', *Security & Privacy Magazine, IEEE*, vol. 3, no. 2, pp. 40-7.

Sahami, M, Dumais, S, Heckerman, D & Horvitz, E 1998, 'A Bayesian Approach to Filtering Junk E-Mail', paper presented to AAAI-98 Workshop on Learning for Text Categorization.

Scanlan, J, Lorimer, S, Hartnett, J & Manderson, K 2004, 'Intrusion Detection by Intelligent Analysis of Data Across Multiple Gateways in Real-Time', paper presented to Australian Telecommunication Networks and Applications Conference, Bondi Beach, New South Wales, Australia.

Scanlan, J, Lorimer, S, Hartnett, J & Manderson, K 2005, *A Context Aware Attack Detection System Across Multiple Gateways*, Unpublished, <http://eprints.comp.utas.edu.au:81/archive/00000085/ >.

Seifried, K 2003, *Information security / TCP Ports list, UDP ports list*, viewed Aug 24 2005, <http://www.seifried.org/security/ports/>.

Spam Prevention Early Warning System, 2005, SPEWS.org, viewed 17 Aug 2005, <http://www.spews.org/>.

The Spamhaus Project, 2005, The Spamhaus Project Ltd, viewed Aug 17 2005, <http://www.spamhaus.org/>.

Stoll, C 1991, *The cuckoo's egg : tracking a spy through the maze of computer espionage*, Pan Books, London.

Trend Micro RBL+ Service, 2005, Trend Micro Incorporated, viewed Aug 17 2005, <http://www.trendmicro.com/en/products/nrs/rbl/evalu ate/overview.htm>.

Wald, A 1947, *Sequential Analysis*, John Wiley and Sons, New York.

Weiss, A 2003, 'Ending spam's free ride', *netWorker*, vol. 7, no. 2, pp. 18-24.

Whitworth, B & Whitworth, E 2004, 'Spam and the social-technical gap', *Computer*, vol. 37, no. 10, pp. 38-45.

Yoshida, K, Adachi, F, Washio, T, Motoda, H, Homma, T, Nakashima, A, Fujikawa, H & Yamazaki, K 2004, 'Density-based spam detector', in *Proceedings of the 2004 ACM SIGKDD international conference on Knowledge discovery and data mining*, ACM Press, Seattle, WA, USA, pp. 486-93.