Real-time Blacklisting of Bots based on Spam Analysis

Wesley Pronk
University of Twente
8106BD Mariënheem
The Netherlands
w.a.pronk@student.utwente.nl

ABSTRACT

In this paper, we describe how analyzing spam can contribute to the identification of bots. This is done by producing a user-defined real-time blacklist based on spam messages. Blacklist analysis shows that blacklisting is not an efficient technique in fighting spam. The analysis also helps in deciding the examination period the blacklist should consider for different thresholds.

Keywords
Botnet, Blacklist, Spam, Filter

1. INTRODUCTION

Botnets are the primary means for cyber criminals to carry out their nefarious attacks [12]. A botnet is a network of compromised computers controlled by a masterbot possibly through several controller bots. Distributed Denial-of-Service (DDoS) attacks, phising, clickfraud, identity theft and sending spam are some malicious applications of such networks.

For botnets, e-mail is an important means of distributing spam. Hence, to enlarge the botnet. Spam is often referred to as Unsolicited Bulk E-mail (UBE), Unsolicited Commercial E-mail (UCE) or junk mail. About 90% of all the e-mail messages sent over the internet is believed to be spam [2]. Nowadays, spam is sent from a large number of bots in small amounts to evade detection.

To obstruct the distribution of spam, most mail servers are equipped with spam filters. Such filters flag incoming messages of which it thinks it is spam and a mail client may place these messages in a so called junk folder. These filters implement several different mechanisms to determine if a message is spam. Current mechanisms can be classified into two categories. The first category identifies and filters spam based on the characteristics of the sending SMTP server, the second based on the message content. Because these two categories are applied at different stages in the process of message acceptance by the receiving SMTP server, they are also called pre-acceptance and post-acceptance tests [6].

2. BACKGROUND

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

13th Twente Student Conference on IT June 20th, 2011, Enschede, The Netherlands.
Copyright 2011, University of Twente, Faculty of Electrical Engineering, Mathematics and Computer Science.

In this section we provide an overview of several techniques used for sending spam as well as techniques for mitigation.

2.1 Spamming

Normally a Mail User Agent (MUA) connects to a Mail Transfer Agent (MTA) to send an email via the SMTP protocol. This MTA forwards the message to another MTA (also via SMTP) until the Mail eXchange (MX) server, another MTA, of the recipient is reached. By now, the recipients MUA can receive the message via POP. Spam message is not sent this way, because the sender wants to stay anonymous. Several spam transmission methods are discussed in the following subsections.

2.1.1 Open Relay

In the past, SMTP servers accepted relay requests from unauthenticated hosts to any other host. Therefore, it was the most common method to send spam. Nowadays, the server's default settings disable open relay requests. Besides that, some ISPs use authentication and whitelists of IPs and domains. Botnets then stopped using SMTP servers and started running their own SMTP servers, which were acting as an open relay. This form of relaying spam can be obstructed by the network in which it resides, if the network manages port 25 traffic as recommended by the Messaging Anti-Abuse Working Group (MAAWG) [1]. They recommend to drop all outgoing mail traffic except from trusted mail servers.

2.1.2 Open Proxy

An open proxy is a proxy server that allows connections from any source host to any destination host. A proxy server is mostly used to stay anonymous, because it alters the IP address of the source. A spammer can send messages from behind a proxy server via the SMTP protocol directly to a MTA or MX server. Sending messages to a MX server in the same domain as the proxy server is called proxylock. In this case, the proxy server has to look up the MX record of its domain. Proxylock has some benefits compared to open proxy. First, the message looks more legitimate, because the proxy is in the same domain. Secondly, the effort to find a relay mail server is decreased. And finally, the message is send in a more distributed fashion, better hiding the original source. This technique also has some disadvantages. The MX record gives the IP address of the inbound mail server of the domain. Since inbound and outbound mail needs different processing, mail servers are often separated. Therefore, sending spam to recipients outside the domain is impossible. The SPF record does give the IP of the outbound mail server, but policies as user authentication and rate limiting are obstructing malicious usage by spammers.
2.1.3 Direct-to-MX
According to the SMTP protocol the MTA queries the DNS for the MX record of the recipients domain. A spammer can deliver mail directly, but it has to query the DNS by itself for the MX record. Direct-to-MX is in favor, because the number of intermediate relays is minimal and thus the number of filters. This way of delivering spam messages brings the potential risk of being blacklisted and if the MAAWG recommendations are applied, direct delivery outside the network is impossible. In addition to that, the spammer has to maintain a list of MX records or query the DNS for the MX record for each domain it wants to send messages to.

2.1.4 BGP Spectrum Agility
Spammers advertise a hijacked IP address range via the Border Gateway Protocol for a short time. This address space is then used to send spam from mail relays and the route is withdrawn from the network. This technique is one of the most sophisticated and it is very hard to trace the spam source. At most 10% of todays spam is sent using this technique [9].

2.2 Mitigating Spam
Filtering is one of the most used anti-spam techniques, which classifies emails based on their content (header or body).

2.2.1 Blacklists
One of the first deployed techniques to block spam were black- and whitelists. Typically, a blacklist is used to block a message or to flag it as spam and a whitelist is used to identify legitimate messages. A traditional static blacklist (or blocklist) is a very basic pre-acceptance mechanism to block spam. Blacklisting is used to block messages from well-known spamming hosts, but it has drawbacks. If an IP address is not (yet) listed on the static blacklist or if it is removed from the list, malicious messages will not be blocked. It takes a certain amount of time and evidence before a spammer becomes listed. Evading static blacklists also is very easy by using the agility of DNS (e.g. using new domains) or using dynamic IP addresses. Furthermore, legitimate e-mails sent from a blacklisted host are rejected. Spammers took full advantage of the drawbacks the traditional blacklist had. Thus, new spam mitigation techniques have been developed based on blacklists.

2.2.2 Distributed Sender Blackhole listing
A Distributed Sender Blackhole List (DSBL) is a DNS based blacklist. DNS listing is a technique for implementing blacklists. These lists have been so widely adopted that the IRTF Anti-Spam Researching Group (ASRG) standardized the structure, the usage and the protocol used to query these DNS based lists [8]. With DSBL, testers abuse insecure mail servers and try to send a specially formatted message to the DSBL system. This message is provided with a time-sensitive secure cookie to prevent false listings. The DSBL system extracts the IP addresses of the specially formatted messages and adds them to the blacklist to block the insecure mail servers.

2.2.3 Greylisting
Greylisting is a technique which places each unrecognized sender on a temporary blacklist. The general thought is that legitimate mail servers will retry sending the mail, while spammer most likely will not. The receiving MTA can refuse a message with a SMTP temporary rejection error code, which can be issued at different stage of the message delivery. If message delivery is retried, the sender is assumed to implement the SMTP protocol correctly and thus to be legitimate. Higher bandwidth usage and CPU loads are the disadvantages of this technique.

2.2.4 IP Reputation
The most scalable and light-weight pre-acceptance mechanism is IP reputation [6]. This mechanism rates senders based on their IP or e-mail address. Based on the reputation of a certain IP address and a threshold, the message is accepted or rejected. In contrast with traditional blacklists, reputation based system can predict the intentions of a host based on a group’s past actions, without any knowledge about the host itself. Since spammers are often found in the same subnet or Autonomous System [7]. DNS lists also have been used to publish IP reputations. Reputation based systems also have some limitations. First, they require huge databases to store the reputation information. Secondly, it is quite easy to evade the systems by changing IP or e-mail address.

2.2.5 Content-based Filtering
These class of spam mitigation techniques classifies emails based on the header or body of a message. Filters are trained to recognize good and bad patterns. Therefore, filtering rules must be updated regularly and large datasets are necessary for training. Since, it is very easy for a spammer to alter a message’s content and thus to protect it from being filtered. Bayesian Analysis is a well known content-based filtering technique and analyzes words and / or phrases in the message’s body to classify messages.

2.2.6 Sender Authentication
Sender authentication is not used to identify spam, but it supposes that an authenticated MTA does not send spam. SPF, SenderID and DKIM are known authentication mechanisms. Sender authentication verifies that a message is sent by the domain it claims to be sent by. The general thought is that a legitimate email is never sent by an email address that does not belong to the domain of the sending MTA.

2.2.7 Challenge-response
"Spam comes from people we don’t know.” is the general idea behind this mitigation technique. To verify whether the sender is a person or a bot, a message is send with a challenge. If the message is responded to, the sender is verified as a legitimate sender. This technique has some major drawbacks. Roughly 40% of the legitimate senders, does not respond to the challenge and are identified as a bot. Secondly, challenges are blocked by many anti-spam systems. At last, spoofing email addresses is an easy way to evade this system.

2.2.8 Distributed Checksums Clearinghouse
The Distributed Checksums Clearinghouse (DCC) is a technique that maintains a database containing the checksums of email bodies and the number of times this message is received. If a checksum is supplied to a server, it responds with the total message count and increments this count. Using DCC can help identifying bulk email. Templates and content randomization, which is relatively easy, evade the system.

There are numerous other spam mitigating techniques, but these are the most common. In this paper we focus on the production of a real-time blacklist. Almost every mail server is equipped with one or more spam filters, but sometimes we still receive spam in our inbox. To help spam filters in identifying those messages,
the idea is to produce a user-defined real-time blacklist based on the headers of spam messages sent to a central server by users.

3. RELATED WORK

Al-Bataineh and White [4] illustrated several methods for the detection of botnet-generated spam and discuss multiple countermeasures to prevent it. Different transmission methods are used by botnets to send spam: relaying, proxying and direct delivery. Two approaches can be taken to detect spam. The network-based solutions try to detect the Command & Control communication channels between a bot and its master. The host-based solutions try to detect malicious activities on the host by monitoring. Several countermeasures can be taken at the network-level, edge routers, and/or at the application-level, the mail servers. Network-level countermeasures are very effective to tackle bots inside the network. The Messaging Anti-Abuse Working Group recommendations prevent spam relay and direct delivery. Monitoring the outgoing mail traffic can give an indication of spamming activities. Countermeasures should take into account the distributed and anonymous characteristics of spam. Countermeasures at many levels and collaboration (between ISPs, administrators, etc.) is needed to obstruct spam. This research gives insight in the methods botnets use to send spam and in the methods to detect and prevent spamming activities.

West et al. [13] developed Preventive Spatio-Temporal Aggregation (PreSTA), a reputation model that uses spatio-temporal information as input to produce a reputation value as output. This value can be used to predict behavior and is useful in partial knowledge situations. In the report, the model is used to detect spam and to prove its effectiveness. PreSTA needs a grouping function and observable negative feedback, in this case blacklists. The hierarchical structure of IP address assignment is used as the basis for the grouping function. For every group there is a reputation value computed based on the negative feedback. The negative feedback used is not a traditional blacklist, but a database consisting of the entries and exits of entities on several traditional blacklists. The computed reputations for each group are evaluated and compared with a threshold to determine if an e-mail message is spam. PreSTA finds 50% of the spam that’s not blocked by blacklists and finds 90% on average when used together with blacklists. The technique overcomes the problem with traditional blacklists, where a bot always has a timeframe to send spam if it is not (yet) listed. In the report the focus lies on producing a reputation value that predicts if the IP address is involved in spamming activities. In this paper an attempt to produce a real-time blacklist of IP addresses that are sending spam will be made. A reputation function can be used to determine if a host should be placed on the list.

Soufe et al. [11] presented a botnet detection system based on the shape of an e-mail. They define the shape of an e-mail as the shape a human would perceive. The analysis of the messages does not rely on the content or the reputation of the sender. They found evidence that spammers used templates to bypass spam filters and used this to block spam messages with the same template. A shape is generated from spam messages and the shape is used by a classifier to identify spam. They were able to classify 100 spam messages with an accuracy of 82%. This method has the same weaknesses as traditional blacklists, spam must be received in order to block future spam messages. In contrast with traditional blacklists, this approach is able to block spam messages created with the same template, but from different origins.

Antonakakis et al. [5] developed Notos, a dynamic reputation system for DNS. The system is based on the fact that agile malicious usage of DNS differs from legitimate usage and has some unique characteristics. Historical DNS information was collected from multiple recursive DNS resolvers and used to learn more about the two kinds of usage. Notos doesn’t produce IP reputation values, but reputation values for domain names. The system was tested in a large ISP’s network with DNS traffic from 1.4 million users. Notos was able to classify new domains with a false positive rate of 0.38% and a true positive rate of 96.8%. The system assigned low reputation values to malicious domains days or even weeks before these domains appeared on blacklists.

IPGroupRep is a reputation based system designed by Zhang et al. [14] to overcome the shortcomings of other reputation based systems. IPGroupRep’s reputation database is more scalable and less vulnerable to be evaded. The difference with other reputation based systems is that IPGroupRep clusters senders based on their IP address and computes a reputation value for each group. The results of the system were compared to the results of other reputation based systems on the same dataset. IPGroupRep is as accurate as the others or more accurate. It effectiveness is the same as the other systems and it classifies less legitimate messages as spam than all the other systems.

Existing blacklists are based on non-persistent identifiers and sending behavior is grouped by domain and not analyzed across domains. Ramachadran et al. [10] developed a system called SpamTracker that filters spam based on the behavior of senders instead of their identity to keep up with the dynamism of IP addresses. The general idea is that spam campaigns, spam lists and spamming techniques are more persistent than IP addresses. SpamTracker uses a clustering algorithm to group the behavior of known spammers based on the set of domains they target. In the classification process is determined how closely related the sending behavior of an IP is with a spamming pattern (or cluster). SpamTracker returns this score to the querying mail server, which can use this score in its existing filtering rules. Behavioral blacklisting can complement existing blacklists and detect many spammers before they are blacklisted. About 10% of the email that is not classified as spam by an organizations filter was detected by SpamTracker.

3.1 Architecture

The system consists of several components, see Figure 1. A Thunderbird (TB) plugin is implemented that sends the headers of selected spam messages a central server, when a user presses the ‘Report Spam’ button. At the server, these messages are processed by a parser. The parser tries to trace the source IP from the headers. If the parser is successful, it inserts the source IP, the date of receipt and the IP of the reporter into the SQL database (SQL DB). All database entries are preserved for examination purposes. The database to DNS-based blacklist (DB 2 DNSBL) script periodically examines the database and places the IP addresses, which are reported more than a certain threshold within a certain timeframe, on the DNS-based blacklist (DNSBL) of a DNS daemon (RBLDNSD). MX servers can query the RBLDNSD to check if an IP address is listed and use this fact in their spam filtering techniques to determine if a message is spam. For example, the result of querying the blacklist can be used as input
3.1.1 Parsing E-mail Headers

A parser is used to parse the spam messages and trace the source IP. To avoid detection, spammers try to obfuscate the real source IP address by forging mail headers. RFC 5321 clearly states that a MTA should prepend its own "Received:" header to the existing ones, but should not alter existing "Received:" headers. The headers consist of a possibly empty list of name/value pairs. Especially the "FROM" and "BY" fields are useful in tracing the source of a message, since they contain information about domain names and IP addresses. These data are derived from client "EHLO" messages and from the TCP connection.

Finding the source involves backtracking and therefore starts at the top most "Received:" header. With the next header we try to verify the previous one. This is done by comparing the IP address derived from the TCP connection information with the results of a DNS lookup for the "BY" field of the header. If there exists an A record for the IP address mentioned in the TCP information with the domain name mentioned in "BY" field of the next header, the previous one was valid. Tracing the mail source has some difficulties. As said before, the name/value list might be empty. This mostly occurs by local mail relaying. We therefore skip the first few headers, if one of these clauses is missing, and try to pick up the trace right after the local relay(s). This local relaying is also possible in the middle of the trace. One could find the localhost address in the TCP information. In that case, you have to go forward (instead of backwards) in the mail trace to find the IP address localhost is referring to. The first address that is not 127.0.0.1 should be used and the trace can be continued. But, if it is a header without the "FROM" or the "BY" field it is deleted.

The "Date:" header is used to extract the date from a message, because we use an existing spam collection and we would like to ‘receive’ spam messages the same way as would happen in a real world setting. This header contains information about when the user actually sends the message, instead of when the MTA processes the message or when the message is delivered. If the header is not present we use the current local time. Because there is no exact date for these messages available and they come from a collection that spans 130 days, they are not taken into consideration when analyzing the blacklist. In Section 3.5 we propose a solution which can be used when testing the setup in a trusted real-world environment with users that are actively involved in reporting spam.

3.1.2 The Database and DB 2 DNSBL Script

The database consists of one table with the following fields:

- an unique identifier
- IP address of the spamming source
- date when the entry was added
- IP address of the reporter
- date when the entry was last updated (unused)
- notes why the entry is listed (unused)
- blacklisted or whitelisted (only blacklist)

In this setup the only fields used to produce the actual blacklist are the IP address of the spamming source and the date when the entry was added. The structure is according to the DB 2 DNSBL script [?], which is an existing script used to create the blacklist based on the database entries. The IP address of the reporter is saved, so excluding abusers is possible. The parser never updates or deletes existing entries, but always adds a new entry after the successful parsing of a header. This way of storage has the advantage of flexibility and simplicity. The blacklist is named created using a certain threshold, but providing public access to the database and letting MX servers perform their own queries gives them the flexibility of determining their own thresholds and time intervals. The notes fields is not used because all those fields would be the same: "Reported by user." This research concentrates on the creation of a real-time blacklist, therefore the whitelist flag is not used. One could delete the unused fields, but they might be very useful in future extensions.

3.2 Research Methodology

For testing purposes and statistical analysis we use an existing spam collection [3]. The collection consists of 151,921 messages collected from January 1st to May 11th.
2011, a period of 130 days. On average more than 1168 messages per day. The owner employed several bait addresses to trick e-mail harvesters. Most messages in this collection contain forged received headers.

Producing blacklists for every hour from 1 January to 11 May and performing statistical analysis on them is not very convenient. Therefore, the analysis is done by querying the database. We have chosen 18 fixed dates with one week between at which the blacklists are examined. This way we can still see how the size of the blacklist would evolve over the 130 days period.

We also had to decide on a threshold. On the one hand, we want the threshold to be as high as possible to minimize false positives and to prevent abuse. On the other hand, it should not be too high in order to effectively list spammers. The system should be used within a trusted environment, so we can weaken the demands for abuse. We therefore chose two low thresholds, 2 and 3. If an IP address is reported more than the threshold within a certain timeframe it becomes blacklisted.

For the timeframe in which the IP addresses have to be reported more than the threshold in order to become blacklisted the same trade-off must be made. Of course we can list every IP that is reported more than 2 or 3 times over a large period, but this increases the number of false positives and the time before the address disappears from the list. On the other side, we don’t want the period to be too small. Then, a very small number of IPs becomes listed. A short examination period till 24 hours with a 3 hour interval is analyzed and a long examination period from till 10 days with an interval of 1 day.

3.3 Blacklist Analysis

We were able to parse and trace more than 99.7% of the messages. From the other messages, 54.6% was untraceable due to timeouts of DNS lookups or a missing "BY" clause in one of the "Received:" headers. And 45.4% of the unprocessed messages contained unparsable "Received:" headers.

From all the spam messages we received and sources we traced a very large number of IP addresses, 73.6%, occurs only once. This number questions the effectiveness of blacklists. In Figure 2 we provide an overview of the IP occurrences.

The results of the analysis over the short examination period are presented in Figure 3. The total number can be seen as a linear function over the examination period. To conclude what the best examination period is we also graphed the weighted average in Figure 4. This figure shows that an examination periods of 12 and 9 hours give the best results with tresholds 2 and 3, respectively.

We did a similar analysis for longer examination periods. The results are shown in Figure 5 and the weighted averages in Figure 6. An examination period of 6 days is optimal with treshold 2 and 7 days with treshold 3.

In Figure 7 one can see the total number of addresses that were listed at the examined dates for a few examination periods. We could interpret that the total amount of spam sent in March and April increased, but the MessageLabs Intelligence Reports [2] shows that it decreased. The cause of the decrease is the dismantling of the large botnet Rustock. A quick look at the spam archive shows that amount of spam received at our server increased in contrast to the total amount of spam in the reports.

3.4 Conclusions

Blacklisting is not an efficient technique in fighting spam, because the majority of the messages is sent from IP addresses that occurred only once in a timeframe of 130 days.

With this system it is possible to construct several blacklists with different purposes. We examined blacklists with for short and long examination periods. It shows that the optimal short examination periods are 12 hours with treshold 2 and 9 hours with treshold 2. And that the optimal short examination periods are 6 and 7 days with tresholds 2 and 3 respectively.

3.5 Future Work

We rely on an existing parser to parse individual "Received:" headers. This parser is able to extract the IP address from the TCP information fields, but is unable to extract the domain name from these fields. If another parser is used or a new parser is written from scratch, which can extract the domain name from the TCP information, one could also try reverse DNS lookups besides the normal DNS lookups to verify headers. This reduces the risk of listing false positives.

The majority of unparsable messages contain headers with local IP addresses, which lacked the "BY" field. So, occasionally some local IPs become blacklisted. A MX server using our blacklist must make sure it doesn’t flag spam messages from a legitimate user by verifying if the IP is non-local. This is not desirable, so the parser should do a final check to overcome the blacklisting of intra network hosts.
Thunderbird is an open source mail client and therefore the source of the plug-in can be easily manipulated and the system becomes an easy target for abuse. We made the assumption the system is used within a trusted environment by reliable users, but one user can still report a message more than once (on purpose or not). The Thunderbird plugin should maintain a list of the messages that were already reported, to ensure a message can be reported once per user.

An inspection of the database showed that some dates are in the future. 12,307 incorrect dates were added to the database. This can be caused by the date parser, but they could also be forged by the spammers. It would be optimal to use the time the spam message was received by the user. Instead of using the "Date:" header, which can be forged, one could also use the timestamp prepended to the last, topmost, "Received:" header. In this way, we extract the delivery time, not the sending time.

4. REFERENCES
Figure 7. Total number of blacklisted IPs for several thresholds and examination periods