Mail filtering on medium/huge mail servers with j-chkmail

José-Marcio Martins da Cruz

Mail filtering on medium/huge mail servers with j-chkmail

José-Marcio Martins da Cruz

Ecole des Mines de Paris – 60, bd St. Michel – 75272 Paris

e-mail:Jose-Marcio.Martins@ensmp.fr

Abstract

Mail filtering on huge servers is a very difficult problem, mainly in regard to efficiency and security. A filter presenting excellent results when running on small or medium servers may present deceiving results when used on huge servers. This paper presents an approach mixing behaviour and content-analysis to scale some filtering methods on huge servers.

1 Introduction

Ten years ago, as soon as Internet really began its large-scale deployment, malicious users immediately understood that this communication vector could easily be used to achieve their goals.

Probably, the first large, malicious usage of Internet was the massive sending of messages to unlimited recipients. In the early days, bandwidth was an issue. So spammers [1] began using open -relays [2]. This was nearly immediately solved both by the generalisation of anti-relaying control on mail servers' software and by the appearance of RBLs [3]. But, in those days, the real big problems were the load imposed to open relays and the amount of non-delivery notifications (bounces) received by innocent senders.

Some time later, viruses appeared, using messaging systems to spread themselves. The new era began with Melissa [CERT99] and LoveLetter [CERT00]. Since then, viruses have been the big threat, as they can really generate irreversible damage to information systems and user data. Usual solutions found for this problem were the generalisation of virus scanners on both users' computers and on mail servers. This was the first real content-filtering feature added to mail servers. Virus scanning on mail servers is a very resource-consuming task. It involves:

- interpreting the message and extracting its on-line content and all attached files;
- examining each extracted part and looking for the presence of virus signatures (usual database size is around 100000 signatures).

Meanwhile, spam activity increased to a very high level. Nowadays, some Internet Service Providers (ISPs) estimate at 90 % the quantity of spam on email traffic over Internet. It is not unusual to find people getting 100, 200 and even more spam messages each day inside their mailboxes.

---

1 spam – general expression making reference to unsolicited electronic message

2 Open relays – mail servers accepting the relay of messages from any source to any destination.

3 RBL – Realtime BlackList: DNS-based list of open relays and spam sources.

2 Medium/Huge servers

By medium/huge mail servers, we're talking about mail servers used by many thousands of users, handling hundreds of thousands connections a day. Mail filtering constraints on huge servers are not the same as those on small and medium servers. Some filtering-related constraints are :

Users Diversity – user profiles on a big university campus may be very diversified: social sciences, economics, computer science, physicians, management ... This diversity implies that one cannot define a typical mailbox: a global typical mailbox doesn't match individual mailboxes and vice-versa.

Scalability – it is easy to build a system handling, say, 50,000 connections a day for 1000 final users. But the goal is that needed computing doesn't grow faster than traffic level.

Surges – huge mail servers will have to have enough spare resources in order to "adequately" handle unattended events, such as bursts of messages or connections.

Human factors – interactions between administrators of servers and final users are simpler in small and medium size organisations: usually they personally know each other. Getting the user-feedback and reliable information needed to tune the filter is much easier on small systems.

Low level constraints – these are the limitations at hardware- or operating system- level, usually forgotten on small systems: the number of processes running on the system, disk I/O bandwidth, network I/O bandwidth, etc.

Site architecture – at big sites, filtering is usually done at gateways, not mail storage servers. Usually, they don't have any information about users other than its existence.

Reliability/Availability – on huge servers, downtime should be as low as possible.

3 Mail Filtering

It is common to classify mail filters in two categories: content and behaviour filtering, but it is not always easy or possible to set up a clear separation between them.

3.1 Content filtering

Content filtering is based on the analysis of data found inside message bodies or envelopes. Methods range from simple pattern matching to complex language processing.

Let us present some of them, but limit the discussion to methods which can be found in j-chkmail.

Pattern Matching - this is probably the most basic filter we can use. The goal is to verify if one or more regular expressions from some defined list may be found in the message. If yes, then the message may be rejected or some score may be assigned to it. The cost of this kind of filtering is very high as each expression has to be matched against the whole message. The filter is unusable if more than some few hundred expressions are defined. Required effort to maintain lists of expressions is very important as, e.g., the same word may appear with many variants (viagra, v1agra, vi@gra, ...). It is very difficult to automate the pattern extraction task if we want, at the same time, to minimise the number of expressions and maximise its coverage.

URL filtering – this method is a variant of the previous, but much more efficient. Arriving messages are scanned once in order to extract all URLs. The domain part of URLs is then looked up at some database. SURBL [JC04] is one of the most effective non-commercial URL databases available: it lists a more than 120,000 domains, its effectiveness is better than 80 % and FP rate [4] lower than 0.5 %. SURBL is available as a DNS zone, but may also be used as a local database (BerkeleyDB format).

Heuristic filters – the most well known heuristic filter is SpamAssassin [SA05]. Each message is submitted to a large set of checks (some hundreds), such as the presence/absence of some headers, html coding quality, the presence of a cryptographic signature, matching mime boundaries fields against some regular expression, etc. Some of these tests fall into the previous two categories.

Each successful test will additively contribute to the message score.

Weights assigned to each test are evaluated in a way to minimise the probability of error in a corpus of messages representing a typical user mailbox. So, 'message score evaluation' is a kind of distance measurement – how far the arriving message is from the user typical legitimate message.

Recent SpamAssassin versions removed most checks with a negative score. Checks with both positive and negative weights are an issue as score-evaluation is not monotonic and all checks need to be performed, even if only few tests are enough to classify the message.

Bayesian filters – The incoming message is broken down into small units [PG02] and a spam rating is then computed for each unit, based on the frequency they appear on the typical user mailbox. Some filters use words as the basic unit, but others filters had found different ways to categorise text. The efficiency of Bayesian filters is usually very high, especially if applied to individual mailboxes or to a homogeneous community.

Although they seem very different, Bayesian and heuristic filters share a common characteristic: they are classifiers. Statistical classifiers learn what the user mailbox is. The efficiency of classifiers is optimal when the incoming traffic perfectly matches the mailbox used in the learning phase. This requirement cannot usually be satisfied on servers handling messages for a heterogeneous community.

If we see the classification process as being a distance measurement problem, we can say:

- Higher distances between spam- and ham-typical mailboxes will result in better classification.
- Sending legitimate messages with characteristics found in a spam mailbox will increase the false positive rate.

We shall remark here that some very reliable filtering criteria, such as URL blacklists, are sometimes included in heuristic filters and the weight assigned to them are evaluated in the process of error rate optimisation. This isn't always the best choice, as this kind of criteria is independent of typical mailbox categorization and, most of the time, may have absolute weights assigned to them, instead of being evaluated by an optimisation process.

3.2 Behaviour filtering

Behaviour analysis tries to detect messages or SMTP clients behaving in a way different from the one found in normal situations. Such deviations may be of many kinds, such as, compliance to usual technical standards or "non-human" behaviour of SMTP clients. Most of the time, behaviour analysis implies analysis over some sliding time window, which

4 FP Rate – False Positive Rate – the rate at which a non-spam message is declared as being spam.

implies saving each connection/message parameter. Time-window size depends on the behaviour being checked.

Connection rate is an example of analysis in a relative short time-interval. Sending messages by a human being is a stochastic process: time interval between two messages is a random variable with exponential distribution. Spam messages sent by robots looks like bursts. Connection rate analysis is done within a sliding time window of five to twenty minutes length. Setting a limit on the connection rate is a simple way of avoiding bursts of connections.

Greylisting [EH03] is an example of behaviour analysis over a longer period. RFC 2821 [RFC2821] specifies that an SMTP client shall retry message delivery after a temporary rejection. So, the idea is simple: when the message arrives for the first time, it is rejected with a temporary error result. If the message is proposed again some time later, after some minimum delay, it will be accepted. This makes the assumption that spam robots do not perform error handling and will not retry later. Greylisting time-window size ranges from some hours to some few days.

RBLs are another example of an even bigger time-window: one day to some weeks. These lists are usually constituted by the IP addresses of computers which were seen sending spam in the past few days – but most of the time, this is also an external filtering criteria.

Nowadays, behaviour checking does not detect too much spam. The reason is that more and more spammers are trying to use armies of zombies [5] to send their messages and disguise their activity. A master zombie controller dispatches to each zombie a message and a list of many thousands of recipients. Lists are created in a way to avoid having too many recipients in the same domain. This way, each SMTP server will see very few connections coming from each zombie, and will not have enough data to do behaviour analysis. Only some external observer with a privileged point of view of all the activity of the zombie will be able to detect the unusual activity of this particular computer.

Either way, behaviour analysis remains useful to detect DoS [6] and other evident deviation of normal behaviour.

4 Scaling mail filtering

Let us consider “filter scalability” as the filter ability to increase the traffic level handled with a reasonable increase in its resources consumption. In other words, resources usage will grow, at most, linearly with traffic level.

Some precautions need to be taken when building a high-performance filter. Some points are related to the filter itself, and others to the MTA. Let us recall some of them.

Learn while working - To achieve scalability, one idea is to use some length of history to decide, if possible, at connection time, if the server will accept the message or not. The sooner the connection is rejected, the less it contributes to the server load: this way, the marginal connection-handling cost decreases with the number of connections already handled for the same SMTP client.

Filter results will be observed for each SMTP client, and stored in memory. To optimise memory usage, j-chkmail uses memory at three levels:

---

5 Zombies – these are computers controlled by spammers to send spam. Usually, they are end user computers infected by some virus allowing someone to remotely install and control applications on it.

6 DoS – Denial of Service

- short history (some minutes) the filter stores some figures for each SMTP client.
- medium history (some hours) – the filter stores some figures for each SMTP client presenting some suspect or bad behaviour.
- long history (some days) – some data about SMTP clients presenting confirmed bad behaviour is extracted from log files and stored in local databases, used by the filter.

These classes of history correspond to some kind of dynamic blacklist management, e.g., "sending messages to spam traps", and "sending messages with high content spam score" are behaviours stored at medium lifetime history. SMTP clients presenting these behaviours are blacklisted for four hours. SMTP clients doing too many connections over a ten minute sliding window are blacklisted for ten minutes and until its connection rate falls into a normal value.

Take the filter decision as soon as possible – content handling is much heavier than behaviour and envelope handling. So, if you can decide what to do during early phases of SMTP dialogue (before DATA command) do not wait. That is to say – do behaviour filtering instead of content filtering, whenever possible.

Compromise between doing well and doing fast – while some filter techniques are very efficient, their cost is too high. High cost methods shall be avoided unless their contribution to the global filter effectiveness is big enough.

Avoid external dependencies – external dependencies are the source of two kind of problem: latency delays (which increases the connection handling time) and vulnerabilities – your system may stop answering if some external resource becomes unavailable.

Close long lasting connections – many misconfigured SMTP clients or spam bots close TCP connection without quitting the SMTP session, leaving servers with useless connections open. For most MTAs, each open connection corresponds to a different process. If the number of processes becomes too high, an SMTP server may present scheduling problems. Generally speaking, all kind of long operations should be avoided, even if they do not consume CPU cycles.

4.1 Efficient content filtering

The data in Table 1 can help us understand what happens on a huge server. The important point about the data coming from this filter is that all filtering checks are done at the same point: after the SMTP DATA command. This ensures that all checks are done and we can compare them. This is not true for filters like j-chkmail, where connections may be rejected at early phases and one cannot know what could happen to these messages if all checks were performed. The data in this table summarises six hours of activity of prolocation.net mail servers, and presents the twenty-five more frequent filtering criteria found on 440K messages.

As we can see, the most efficient criteria are URL blacklists, IP blacklists and Bayesian filtering. Heuristic criteria appear less frequently and come from non-reliable checks (HTML message). The very most effective criterion is URL filtering. Pattern matching appears once with very low hit-count, but is probably a good spam indicator.

Another important point to note is the existence of external dependencies: fifteen blacklists found among the twenty-five top hits. This point shows, for this filter, how important external sources of data are. To minimize server-dependability from external factors, a local copy of these lists should be used (as is the case here).

Rank	Count	Class	Criteria
# 1	127535	B	URIBL_WS_SURBL
# 2	127101	B	URIBL_SBL
# 3	125917	B	URIBL_JP_SURBL
# 4	120728	B	URIBL_OB_SURBL
# 5	96849	C	BAYES_99
# 6	95827	A	RCVD_IN_BL_SPAMCOP_NET
# 7	90406	D	HTML_MESSAGE
# 8	71017	B	URIBL_SC_SURBL
# 9	46927	D	MIME_HTML_ONLY
#10	36806	B	URIBL_AB_SURBL
#11	33822	A	RCVD_IN_XBL
#12	30930	D	MIME_BOUND_DD_DIGITS
#13	30649	D	MPART_ALT_DIFF
#14	28472	B	URIBL_AH_DNSBL
#15	26638	A	RCVD_IN_SORBS_DUL
#16	26621	E	DRUGS_ERECTILE
#17	26394	D	MSGID_FROM_MTA_HEADER
#18	24615	A	RCVD_IN_DSBL
#19	23977	D	MSGID_FROM_MTA_ID
#20	23690	A	RCVD_IN_SORBS_SPAM
#21	22457	A	RCVD_IN_NJABL_DUL
#22	21115	A	RCVD_IN_NJABL_PROXY
#23	21013	A	RCVD_IN_SBL
#24	20262	D	X_MESSAGE_INFO
#25	18044	D	HTML_FONT_BIG

Legend

A	IP blacklists	8 criteria
B	URL blacklists	7 criteria
C	Bayesian filtered	1 criterion
D	Heuristics	8 criteria
E	Pattern Matching	1 criterion

Table 1 – The twenty-five most frequent filtering criteria (domain prolocation.net)

From this, a good strategy for content filtering is:

- URL filtering is very effective and will be a primary content filtering criterion.
- It makes no sense to evaluate, with extreme precision, weights assigned to heuristic criteria if a typical mailbox cannot be defined. And, as they are not either highly effective or reliable, this heuristic filtering will be a secondary method. Weights assigned to checks may be based on an estimation of the severity level of the deviation.

- Whitening checks will be minimised. Most of the time, whitening criteria on filters may be abused by spammers, and good whitening criteria are not usually the same for all users. Message whitening is left to recipients, based on their address book. Some level of false positives may be accepted if they can be easily identified by recipients.
- Filter efficacy's is better measured by user's ease of message classification – subjective criteria.
- Regular expressions and URL filtering methods generates very few false positives, if correctly configured, as they represent patterns only found on spams and not on hams. Weights assigned to these checks may be high enough to trigger spam.

Heuristics filtering basically checks message compliance with respect to RFCs and to check some characteristics frequently appearing on spams. There are only 32 tests of this kind on current j-chkmail release and this number is falling.

4.2 Combining content and behaviour filtering to achieve scalability

Behaviour/content co-operation may appear in both directions. When the score of a behaviour check is high but not enough to reject the connection, it can set up some initial score for the heuristic content filter.

On the other hand, gateways sending messages with high scores will have this information stored inside medium term history and will reduce behaviour thresholds applied to these gateways.

If co-operation appears very interesting, care shall be taken to avoid closed loops, in which case the filter may become unstable or starved.

The most interesting case of co-operation between content and behaviour filtering comes from greylisting.

5 Adaptive Delay Greylisting

Greylisting is the last filtering method added to j-chkmail and its implementation is a very interesting example of co-operation between filtering methods. In its basic version, it presents excellent filtering results but its scalability is limited. There are two problems with basic greylisting:

Database size grows with recipient rate, not connection rate. Our tests were validated on a gateway handling around 500K connections per day. Normal database size for this gateway is around 600K records, but we have seen some peaks of 1M records. Grey databases need periodic scanning to remove old records and, given the size of databases, access times may become prohibitive on huge servers. Some filters use disk-based databases (relay-delay [RD03], milter -gris [MG04a] and j-chkmail), while milter -greylisting [MG04b] uses a linked list (in memory) to store the greylisting database;

Database poisoning attacks are possible on greylisting filters. A malicious remote user may be able to fill up the filter database if he does lots of connections and tries to send messages to many recipients from random senders. It is enough to try to send a thousand messages to the same thousand recipients with a different sender each time to create a million entries at the server side greylisting database.

Milter-greylisting partially addresses the first issue: whitelisted entries may use only a client IP address part of the triplet if configured to.

Adaptive Delay Greylisting tries to address these two issues.

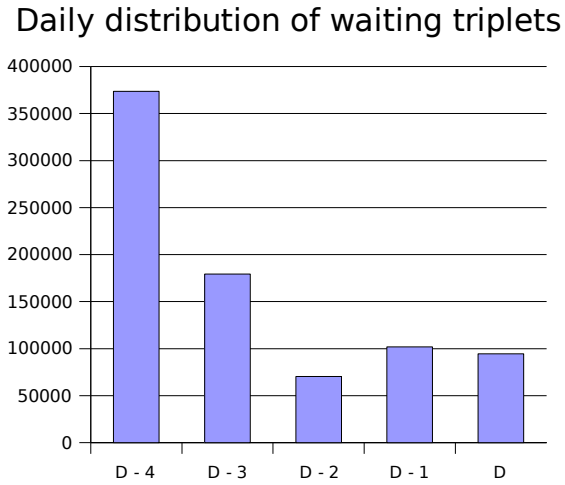

Figure 1 – Daily distribution of the number of triplets in the waiting database.

Figure 1 shows a sample of the daily distribution of waiting entries at jussieu.fr mail gateway. This example shows two probable data poisoning attacks on the first two days.

The first question we can ask ourselves is: how many entries are validated after they remain waiting for more than a specific time? An easy way to answer this question is to sample the waiting database at some specific time intervals (say six hours), create the time distribution of entries and superpose results from different samples. This way, we can have an idea of the number of entries being validated.

Data from domain ensmp.fr shows that the number of waiting triplets validated when they are older than twelve hours is always smaller than 1 %. We can interpret this result as the usefulness of database records : 99 % of the number of entries older than twelve hours is useless.

Another, much more conservative evaluation makes the assumption that all entries older than one day are useless. With this point of view, we can consider up to 80 % the of database records are useless when the maximum waiting time is five days.

The above analysis allows us to set up higher and lower limits on our goals. Even the lower limit is very interesting for database sizes with greater than 500K entries.

So, the question that adaptive delay greylisting tries to answer is: how will we select the records to be removed?

Basic greylisting filters use three time constants [EH03]: the minimum delay to accept waiting triplets, and the maximum lifetime for waiting and whitelisted triplets. The basic idea of Adaptive Delay Greylisting is that time constants are not fixed but depend on some "quality" score assigned to the triplet. There is no reason for, e.g., bounce triplets to have the same lifetime than normal message triplets, or triplets from SMTP clients sending virus or spam to have the same lifetime as normal message triplets. Also, why should the triplet 192.97.20.2!joe@terena.nl!joe@ensmp.fr (erasmus.terena.nl) be handled with the same priority as 209.67.208.34!joe@terena.nl!joe@ensmp.fr (34.208.67.209.reverse.layeredtech.com) ? In both cases, the sender is someone in the domain "terena.nl", but only the first one come from a IP address inside that domain.

Quality score comes from three sources:

- **Greylisting database entries** – data inside a single record may determine if its lifetime will be shortened or not. Examples of data taken into account are: the message is a bounce, the SMTP client address doesn't resolve or the reverse/direct resolutions do not match, or the SMTP client host name and message sender domain name do not match.
- **Greylisting database as a whole** – correlating entries from the same source or analysing how old entries are dropped may be enough to detect well or badly behaving sources – and allow management of simple black or white lists. Database poisoning may be avoided if a limit is set on the number of recent waiting triplets generated by the same source. Examples of criteria are : the number of waiting triplets from this source and the number of different domains coming from the same source.
- **Cooperation with external filters** – recent whitelisted or waiting triplets from some SMTP client may be removed if content filtering applied to recent messages from this SMTP client results in spam most of the time. Examples of criteria are: the mean spam score of messages coming from this SMTP client in the past or the number of viruses coming from this source in the last few hours.

An adaptive Delay Greylisting data flow schema is presented in figure 2. Two databases are added to original greylisting schema: white- and black-entries databases. Valid entries database replace the original white entries database. The lifetime for both valid and black entries may be lowered down to a week. White entries are generated from single or multiple valid entries with very good behaviour. On the contrary, black entries come from waiting entries with very bad behaviour and may be used by other filtering methods.

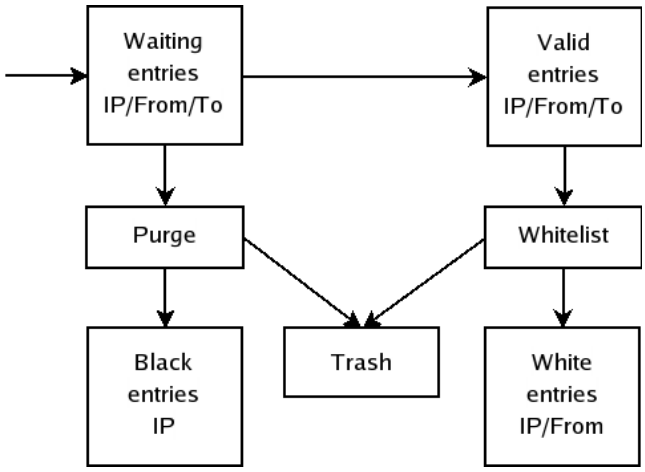

Figure 2 – Adaptive Delay Greylisting data flow

At the time of writing, only results for database cleaning -up based on information internal to the database itself are available. Co-operation with content and behaviour filtering is implemented but the algorithms have not been validated sufficiently.

Results below considers the lifetime of entries are reduced the same way, no matter which reducing criterion is matched, but this isn't necessary. The action applied to the entry may also vary depending on which reducing criterion is matched : while some criteria may generate a reduction on the entry lifetime, others may immediately delete the entry from database (e.g., a virus found on a recent message accepted from this SMTP client).

An interesting presentation of results is given in table 2. This table shows the daily distribution of the number of waiting triplets in the database, when the lifetime of suspicious triplets is reduced to 6, 12 or 24 hours. In all cases, the number of waiting triplets is limited to 1000 for each SMTP client.

We can see from this table that the size of the database may be drastically reduced even if the lifetime of suspicious triplets is set to 24 hours.

	Original	**6 h**	**12 h**	**24 h**
D	75634	25008	36263	71074
D - 1	88432	5216	5216	5216
D - 2	94555	6773	6773	6773
D - 3	101710	6469	6469	6469
D - 4	70304	5460	5460	5460
Total	430635	48926	60181	94992

Table 2 – Reduction results of a waiting triplets-database: daily distribution of the number of records against the lifetime of suspicious triplets

Table 3 shows how valid database size is reduced when its useless content is discarded and its useful content is distributed over itself and white database. Note that 7967 entries from the valid entries database are converted into 1326 entries on the white entries database, as this last one stores only IP/From information.

	Before	**After**
Valid entries database size	22530	4407
White entries database Size		1326
Entries removed		10156

Table 3 – Reduction results of a valid triplets-database

We have seen that Adaptive Delay Greylisting allows reduction of database size. But what would be the influence over other parameters, mainly spam detection and error-rate?

False positive and message loss rate may increase, as it may happen that legitimate waiting triplets are removed from the waiting-entries database before they come back. But in this case, when they come back, another greylisting cycle will be started. If, even after this new cycle, the client cannot deliver its message, it is reasonable to think that he is suffering from some scheduling problem (other than the initial reason the triplet was removed).

False negative rate will decrease. In traditional greylisting, false negatives may happen when some waiting triplet is validated some days later by some message which is not the initial one. As Adaptive Delay Greylisting removes suspicious entries, this situation will become less frequent than before.

6 Results

6.1 Results of behaviour filtering

Results of behaviour filtering are more difficult to evaluate. In a normal operation, the ratio between spam blocked by behaviour filtering itself (no external information sources and no greylisting) and content filtering itself varies between 10 and 20 %. So, it does not really contribute too much to reduce the server-load under usual conditions. This can be explained by the fact that much spam is sent by zombies and behaviour filtering isn't very effective against this kind of SMTP client.

On the other hand, behaviour filtering remains very interesting to block surges of spam or surges of viruses. At ensmp.fr domain we estimate that, when the MyDoom virus begun spreading, at least half of incoming viruses were blocked by connection rate control.

Figure 3, below, shows how two bursts of 20 K connections done in ten minutes are smoothed for specific clients. It is remarkable that as long as the connection rate limit is done on a per-IP-basis, all connections from other sources arriving within this time interval were not disturbed.

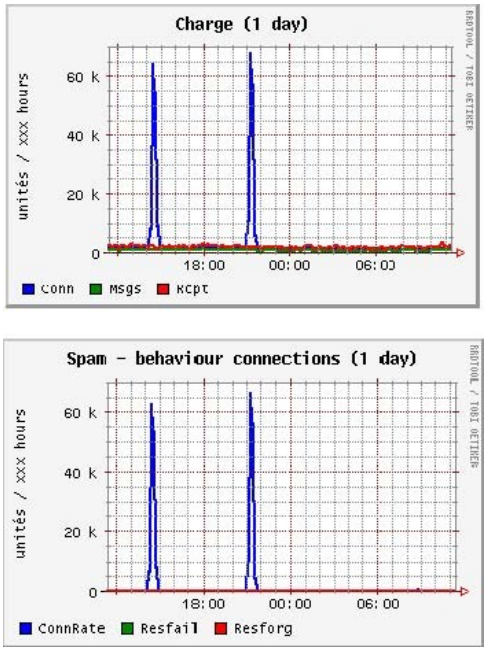

Figure 3 – Connection rate control in action: incoming and rejected connections. Bursts of connections are rejected without disturbing normal traffic

6.2 Results of content filtering

These results, presented in table 4, come from all messages received by the author over fifteen days. This test was done as follows: messages arriving are sent to the normal user mailbox and a copy is redirected to a test the account in an IMAP server using a sieve filter to redirect them to one of three mailboxes. Filtering is done after greylisting.

	Inbox score = 0	SCORE-LO score =[1,3]	SCORE-HI score > 3	Total
Total	2100	218	790	3108
ham	2034	93	31	2158
spam	66	125	759	950

Table 4 – Results of content filtering classification over fifteen days

A score greater than zero means that the message matched some spam check criterion.

These results need some interpretation. This data was collected from the author's mailbox. The author is a computer scientist, and his mailbox-type may not match other profiles which will surely give different results.

The results above are not as bad as one would think, as they result from the "heavy" filtering on the server. All ham messages, whose score is greater than 0, come from discussion lists (some of them are related to spam filtering) and can be pre-filtered (add a personal criteria: he knows the sender), with an algorithm such as:

```
if sender is known then
   put message in Inbox
else if score > 3 then
   put message in SCORE-HI
else if score > 0 then
   put message in SCORE-LO
else
   put message in Inbox
endif
```

Most false negatives come from 419/SCAM messages which are difficult to filter without high false positive rates.

Results are worse than those we can obtain with Bayesian filters, but we shall note that this result is scalable as no assumption was done about the categorisation of the user's mailbox, and on the other hand, this content filtering was applied to messages which had already passed a greylisting filter (spam/ham ratio is biased).

Either way, one can see the efficiency of the filter as the difficulty the final user has to correctly class his mailbox. In this case, if his MUA performs the pre-filtering algorithm shown above, he will need to correct only 4.4 messages a day, which seems to be acceptable.

6.3 Server Load

This goal was fully attained: j-chkmail is a filter which does not consume too many system resources. Table 5 presents some typical values of load. Note that the much bigger consumption of memory with Linux. The reason is the implementation of threads with Linux and the memory allocated, which grows very fast with the number of threads. This is already reduced by using an alternative implementation of libmilter [JM03], based on a pool of workers instead of having one thread per connection.

Domain	OS	Messages / d a y	CPU load (typ)	Memory (typ)
pobox.sk	Linux	350000	< 3 %	140 MB
jussieu.fr	FreeBSD	400000	< 5 %	30 MB
ensmp.fr	Solaris	60000	< 5 %	30 MB

Table 5 – Typical load figures

7 Conclusions

Mail filtering on huge servers is difficult. While it is easy to do reliable filtering for small or homogeneous medium communities, very few filtering techniques are ready to fill the task reliably.

j-chkmail implements some ways to handle important traffic levels, but does not achieve efficiency and the low error-rates found on well-tuned personal filters.

However, j-chkmail is a convenient solution if we accept a reasonable goal which is more qualitative than quantitative - ease of classification of messages by the final user instead of some very high numeric measurement of efficiency. This is possible if users accept finding, from time to time, some spam messages among their legitimate messages. It is more a human than a technical problem.

If we were to select the more effective features implemented in j-chkmail, we could say:

- Connection rate control (and other similar controls) do not block too much spam, but are very interesting to protect the server against unattended traffic surges.
- Greylisting: at least for the moment, this is a very interesting technique as it blocks most spam, with very few false positives. Ideas presented in this paper help with the scaling of greylisting.
- URL filtering – The main criterion used by Surbl.org to manage its URL blacklist, “If it appears in HAM, do not list it” makes it very interesting as it does not depend on a typical user mailbox – a big problem at important organisations. Surbl.org is, at the same time, fast, efficient and reliable.

8 Thanks

Many people contributed to this work, but the author wants to thanks those who specifically contributed in many ways to aspects related to mail filtering on huge servers : mainly Tibor Weis (pobox.sk and tuzvo.sk) and Sebastien Vautherot (jussieu.fr).

But also Jeff Chan (surbl.org), Raymond Dijkxhoorn (prolocation.net and surbl.org), Dennis Peterson, Christian Pelissier (onera.fr) and Serge Aumont (cru.fr).

9 Author

Jose-Marcio Martins da Cruz -holds an Electrical Engineer degree from ITA (Brazil) and a MSc in Computer Science from INT-Evry (Institut National des Télécommunications). He works at Ecole des Mines de Paris and his main interests are mail filtering, security, network monitoring and intrusion detection.

10 Bibliography

[CERT99] “Melissa Macro Virus” - http://www.cert.org/advisories/CA 1999 -04.html

[CERT00] “Love Letter Worm” - http://www.cert.org/advisories/CA2000 -04.html

[EH03] Evan Haris - “The Next Step in the Spam Control War: Greylisting” - http://www.greylisting.org/articles/whitepaper.shtml

[JM03a] Jose M. M. Cruz - “Le filtrage de mail sur un serveur de messagerie avec j-chkmail” -JRES2003

[JM03b] Jose M. M. Cruz – http://j -chkmail.ensmp.fr/libmilter

[MG04a] “milter-gris Web Site” - http://www.milter.info/sendmail

[MG04b] “milter-greylisting Web Site” - http://hcpnet.free.fr/milter greylist/

[PG02] Paul Graham - ”A Plan for Spam” http://www.paulgraham.com/spam.html

[RD03] “RelayDelay Web Site” - http://projects.puremagic.com/greylisting

[SA05] “SpamAssassin Web Site” - http://spamassassin.apache.org