The successful low-cost deployment of a secure Email gateway

Christophe Gaboret[1]

Institut National des Télécommunications,
Computer Science and Audiovisual Engineering Department,
F91011 Évry cedex, France

Abstract. Communicating by email has become crucial for all companies today. However, a significant amount of undesirable messages pass through mail servers. Such unwanted and unsolicited communications are responsible for a significant loss of time and money for all corporations and may be damaging for corporate reputations. We present here the email delivery solution we have implemented in our institute, to prevent users from receiving spam and viruses. Our choice was motivated by many criteria including reliability, availability and security but also flexibility and price. Indeed, this email gateway runs on a Linux server and is based on a few modular open source tools designed for security purposes. We present all the tools involved in this gateway and discuss their advantages and drawbacks. Technical explanations of system optimization and Bayesian filtering are also demonstrated in this paper. Finally, we show the rate of success we obtained on deploying this email gateway.

1 Introduction

The development of the Internet is a reflection of the insatiable need that human beings have to communicate with one another [7] and email has thus become the widely used application on Internet . Sending an electronic message is probably now the most common and widespread gesture of Internet users. Today, this trend is verifiable in all modern enterprises and impacts employees, customers, partners and suppliers.

Communicating by email is crucial and constitutes essential conduit for exchanging internal and external information. Therefore, any perturbation or even a suspicion of a problem is generally little appreciated by users. For this reason an email delivery solution should be reliable and 100% available.

The success of email has of course brought with it a number of problems which include Internet worms, viruses, spam, phishing and fraud. The result has been a significant loss in productivity, time and money for all businesses and may be also damaging for corporate reputations. An effective email delivery gateway should therefore be able to protect users against all these threats.

In this paper, we present the successful deployment of a secure email gateway, at the *Institut National de Télécommunications* (INT). The email delivery solution we have implemented is able to detect viruses, spams and similar attacks, and has proved itself extremely effective and safe. We wish to emphasise that because telecommunications are our core activity it is of paramount importance that the department provides a working solution for all communication needs and this, in an extremely heterogeneous environment which includes major research projects, teaching and administrative staff and well over a thousand students.
After a brief explanation of the underlying protocols, we describe our initial email gateway and describe our needs. Then, we discuss the choice of Mail Transfer Agent (MTA) and the different components involved in detection of viruses, spam and other security attacks. Finally, after a few technical indications in particular dealing with optimization, further insights are given.

2 Project specifications and system requirements

2.1 A few words about email protocols

Email allows any user connected to Internet to send messages. The transport of an email between the different sites around the world is usually made by the protocol SMTP (Simple Mail Transfer Protocol). In other terms, SMTP is the application-level protocol that handles message services over TCP/IP networks. SMTP was defined in 1982 by the Internet Engineering Task Force (IETF) and is currently specified in RFCs 821 and 822. SMTP uses TCP port 25.

Typically, when sending a message, a DNS (Domain Name Service) request is done to discover how to reach the Mail eXchanger (MX) of the recipient. If an MX is found then the message is transferred to this server, through port TCP/25.

Although SMTP is the most prevalent of the email protocols, it lacks some of the features. For example a primary weakness of standard SMTP is the lack of support for non-text messages. MIME (Multipurpose Internet Mail Extensions) supplements SMTP and allows the encapsulation of multimedia (non-text) messages inside of a standard SMTP message. Some mime types as MS-TNEF[1] (Microsoft Transport Neutral Encapsulation Format), which is a proprietary standard from Microsoft, are more specific and need relevant tools to be interpreted correctly.

2.2 Discussions about our initial email gateway

Distribution of email has exists for years in our institute but few modifications have been introduced. At the beginning of this study, our email gateway was composed by a heterogeneous succession of three servers: an initial Linux server

[1] The MS-TNEF is usually generated by Microsoft's mail client either Microsoft Outlook or Exchange Mail. The format encodes the attributes of messages such as fonts, colour of fonts or type face of fonts.

running *Sendmail* in a DeMilitarized Zone (DMZ), then a Windows server with a virus scanner and finally another Linux server containing all the mailing addresses.

This meant relatively complex email processing with many stages and therefore many potential places where a problem could appear. In fact whenever a message came from another site, it was sent to the first Linux server. Then, the Windows virus scanner received this email and worked on it. After, this message was sent again to the first Linux machine. Finally, the mail arrived on the Linux server which hosted the mailboxes.

We noted two main sources of trouble. First, the server with the electronic addresses was a potential single point of failure because all the traffic depended on its availability. Second, the virus scanner hosted on the Windows server was the same one that was on the client's desktop and therefore a malicious program which remained undetected could infect the user's machine. Furthermore, our virus scanner was a US product and definitions were delivered with a time lag of about 6 hours which could be dramatic in case of a European virus attack. Finally, not only was this system not able to filter spam, but it was also quite expensive (2000 purchasing price and around 400 a year after that). Nevertheless, users were familiar with the program and its reputation was reassuring.

2.3 Required specifications

The INT possesses approximately 2000 mailboxes, and around 10 million email pass through its SMTP servers every year. There is a significant volume of traffic. Traffic volume is significant, in addition many users have mailing addresses on external websites, so our choice was between installing on one powerful server or on two servers.

The National Institute of technology owns approximately 2000 mailboxes, so about 10 millions emails pass through it SMTP servers. Volumetry of traffic is significant, as well as many users leave their mailing address on many website, so our future gateway should be implemented at least on a powerful server.
As email is fundamental for all users, especially in the INT, we would prefer a round-robin (DNS load balancing) system of two servers. Then, if one server crashes, the second one could take on the mail delivery.

For most users, email is an essential work-tool: everyone should have the possibility to send or receive a message at any time of the day, 365 days of the year. This is particularly true for researchers who need to exchange information with international colleagues. Consequently, availability was a determining factor in our choice of solution. This required that the servers which make up the gateway be secure and every effort should be made to investigate any factor that compromised that security.
Because, many clients run different Operating Systems (OS) the SMTP server which hosts our future gateway should have the ability to recognise all formats (see above 2.1). As most INT users (around 80%) work on Windows systems, it

might be more convenient to host this gateway on the same OS. However, in our educational environment, the costs involved are critical.
Open Source software is free of charge and more and more compliant with proprietary formats. Furthermore, we are convinced that the only way to achieve the required level of confidence and security is to employ Open Source solutions. Indeed, commercial suppliers can not compete with thousands of developers specialising in security.

French and European law stipulates that everyone has the right to privacy in his or her private life and correspondence. (Article 8, European Convention for the Protection of Human Rights and Fundamental Freedoms). An employer can not look at employee's email even his computer belongs to the company and even if the rules of use mention this possibility [1]. It is therefore necessary for the department to employ automatic "blind" technical solutions in order to analyse message content.

To recapitulate the different points evoked above, our email gateway specifications are to:

- Use automatic analysis of the email
- Reduce the number of servers involved in order to streamline the treatment of mail
- Favour a Linux solution to simplify the delivery process
- Employ an Open Source solution to reduce the cost
- Use both free and commercial virus scanners to reassure users
- Detect, filter and sort undesirable messages
- Hide the aliases lists from outside of our domain
- Make transparent changes for the users with no messages lost

In other terms, we wanted to put into place a straightforward, open, low-cost, secure, efficient and reliable email gateway solution.

3 Secure mail gateway software

To scan all email for viruses, spam, phishing and other malicious programs, software is needed. To securely perform this role, an Open Source toolbox is required. , In addition to its low cost an Open Source gateway allows the possibility to incorporate different commercial virus scanners as well.

Two software programs present the features mentioned above: Amavis and MailScanner. When we started our assessment, Amavis did not offer the same flexiblity as its rival so we only present MailScanner here, which is the software we now employ at the INT.

3.1 MailScanner

MailScanner provides the engine used to scan email, detect security attacks, viruses and spam. By virtue of being Open Source, the technology in MailScanner has been reviewed many times over before becoming a reliable and trustworthy solution. This software is written in Perl for three main reasons [12]:

- Using Perl eliminates all the memory allocation and buffer overrun problems associated with something like C
- Only parts of the process are CPU intensive
- Perl is far more "portable" than most other languages, so it can be run on almost any platform

MailScanner is used by over 30,000 sites around the world, protecting top government departments, commercial corporations and educational institutions. This technology is believed to become the standard email solution at many ISP sites for virus protection and spam filtering [4].
This software can be found at `http://www.mailscanner.info`

The MailScanner engine initiates email scanning by starting two instances of the Mail Transfer Agent.
The first MTA is started in daemon mode to accept incoming email. The message is accepted and simply delivered to an incoming queue directory. To accomplish the scanning of incoming emails and processing tasks, MailScanner starts a configurable number of child processes. Typically, there are five child processes which examine the incoming queue at five second intervals and select a number of the oldest messages in the queue for batch processing[2]. MailScanner processes the waiting message and then delivers the cleaned messages to the outgoing queue directory. Only after the messages are delivered to the outgoing queue directory are they deleted from the incoming spool directory. This ensures that no mail is lost, even in the event of unexpected power loss, as the system always has an internal copy of all messages being processed.
The second MTA instance is also started in daemon mode and watches an outgoing queue directory for scanned and processed messages that need to be delivered [11].

In other terms, MailScanner is not involved in providing SMTP service, or delivering emails but this software sits between the two instances of the MTA (*i.e.* between the two queues), moving mail from the incoming to the outgoing as it scans it (see fig. 1 for explanations of the process flow).

[2] The number of child processes and the time interval between them is configurable and should be established based on the gateway system's speed, memory, number of processors and other running applications.

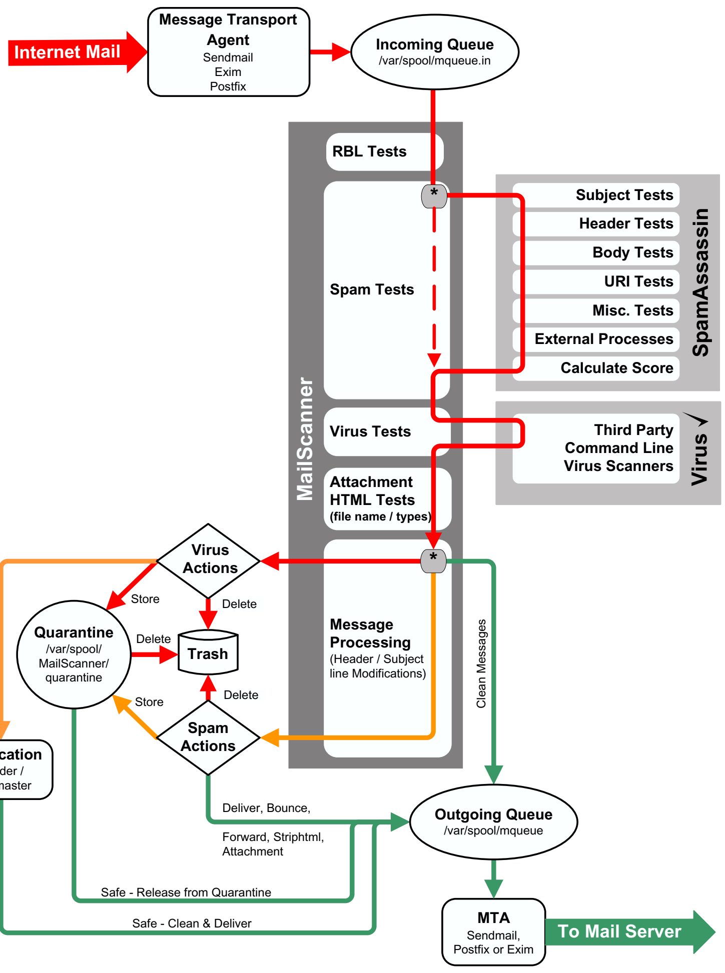

Fig. 1. MailScanner process flow, (courtesy of J. Field). We see that MailScanner runs between two instances of the Mail Transfer Agent to accomplish scanning of emails in order to detect security attacks, viruses and spam. MailScanner first runs a series of Real-time Black List (RBL) tests on each message. If the message passes successfully the RBL tests it is passed to SpamAssassin which uses heuristic, Bayesian and other tests to determine the spam level of the message (SpamAssassin assigns a numerical value to each test that is used on the message). Every message receives a “spam score”. Then, a virus scan is performed using related scanner(s): if a virus is detected, the message is marked as containing a virus. Once virus detection is complete, the MailScanner child process examines the filename and file type of any email attachments against site configurable rule sets. Virtually any type or name of attachments can be blocked or passed depending on how MailScanner has been configured. The message is also examined to see if the body contains possibly dangerous HTML content. Configurable options allow logging, passing, deleting or disarming these HTML content tags. After this stage of the processing, MailScanner has all the information needed to modify, deliver, reject or quarantine the message. This final message processing depends on the message content and the MailScanner configuration settings [11].

3.2 Choice of the Mail Transfer Agent

A few MTAs exist like *Sendmail*, *qmail*, *Exim* or *Postfix*[3]. Each of these four widely-used MTAs has broadly similar features. They can all handle large amounts of mail and can interact with databases in many formats. They have an extensive knowledge of the many SMTP variants in use and are not readily exploitable. The source code is freely available along with third party document support.

Sendmail is the most popular MTA and is reckoned by many authorities to deliver a bit less than half of all Internet email. That works out to be billions of messages every day. Sendmail is one of the main reasons that the Internet is useful to many millions of people, and certainly sendmail is how Internet email was developed in the first place.

Sendmail has an extraordinarily obscure configuration file, a poor history of security breaches and a design centred around UNIX in the early 1980s. It is a fact that hundreds of thousands of sendmail sites are currently advertising themselves as having remotely exploitable security vulnerabilities.
The primary design goal for qmail is to replace some Sendmail features, giving more security and performance in the process.
The outstanding feature of Exim is that it was designed to be a general-purpose mailer for UNIX machines. Exim is not a total rethink about how mail works, like qmail is. Exim looks and behaves much like any other UNIX daemon, with a monolithic configuration file, a monolithic daemon, a small number of log files and a standard style of spooling. It does not have a poor security history, can cope with high load and it has excellent integration facilities.
Postfix is, like qmail, written by a prolific freeware security specialist. Postfix fits somewhere between qmail and Exim. It consists of several programs (but fewer than qmail), and has a substantial configuration file. Postfix has a strong emphasis on security, but not to the extent of imposing unusual UNIX management practices. Postfix is quite flexible in its configuration file, but not to the extent of Exim (Exim was designed to be a general-purpose mailer for UNIX machines). Postfix has been measured by many as being extremely fast. Postfix is, like Exim, a drop-in replacement for Sendmail.

Our MTA selection criteria are:

– Ease of administration
– Security
– Performance
– Long-term viability

Actually, all the cited MTA are comparable but for either security reasons and because we have a number of years experience of Postfix (internal competence), we chose this MTA.

[3] There are other commendable MTAs one can talk about, such as zmailer and smail3 (not as widely used) or products like Microsoft Exchange or Lotus Notes but we decided to omit them.

Postfix is a complex system, running with reduced rights and privileges, using separate independent processes. It does not run under control of a user process (controlled environment), it is not a set-UID program (*i.e.* able to write and to change rights). Postfix programs do not trust the contents of queue files (queue files have no on-disk record for deliveries) and the number of in-memory instances is limited while the memory for strings and buffers is allocated dynamically in order to prevent buffer overrun problems. For the same reason, large inputs are broken up into sequences of reasonably-sized elements.

Most Postfix daemon programs can be run at fixed low privilege in a chrooted environment. This is especially true for the programs that are exposed to the network: the SMTP server and SMTP client. Postfix uses separate processes to insulate activities from each other (see fig. 2 or details). In particular, there is no direct path from the network to the security-sensitive local delivery programs. Some parts of the Postfix system are multi-threaded. However, all programs that interact with the outside world are single-threaded. No Postfix mail delivery program runs under control of a user process. Instead, most Postfix programs run under control of a resident master daemon that runs in a controlled environment,without any parent-child relationship to user processes. This approach eliminates exploits that involve signals, open files, environment variables, and other process attributes that the UNIX system passes on from a possibly-malicious parent to a child. Postfix queue files have a specific format; less than one in 10^12 non-Postfix files would be recognized as a valid Postfix queue file. Postfix programs do not trust data received from the network. In particular, Postfix filters sender provided data before exporting it via environment variables [2].

If Postfix uses multiple layers of defence, it is precisely because this program was written with security in mind. That is why, architecture of these multiple programs are so difficult to break [10].
Postfix can be found at `http://www.postfix.org`

3.3 Treatment of spam

Spam is unsolicited and undesirable email sent generally to sell a product such as software, pharmaceutical products or encourage access to a pornographic sites [6].

The number and variety of these types of messages has increased dramatically. Today, the amount of spam sent to a site could even create bandwidth saturation. As a result other applications could have difficulties to gain external network access [1]. As well as monopolising computer resources much time is wasted in dealing with it. Classic filtering techniques which drop messages containing key words like *viagra* are not sufficient. Indeed, spam with a spoofing, for instance misleading subject headers, are commonly seen [5].
A recent study revealed that the average cost of spam for an American company is more than $150 per year and per person [3] and it may double over the next two years.

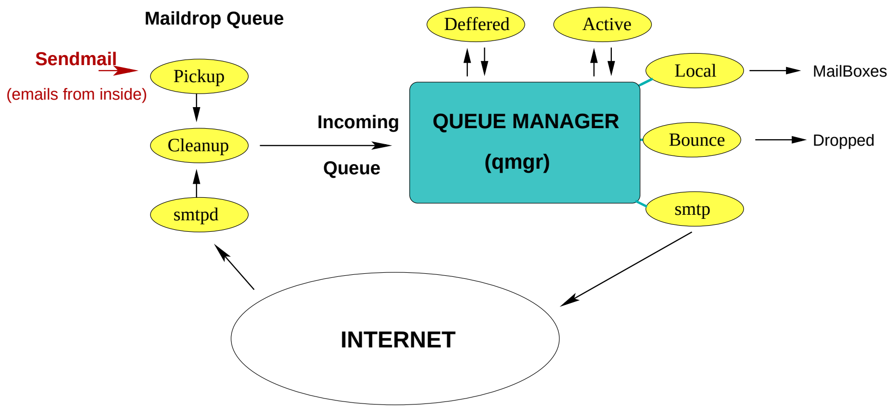

Fig. 2. Simplified overview of postfix mailing system: Any messages from inside our domain are transferred by the sendmail emulator into the maildrop queue. The pickup daemon reads from the maildrop queue and puts the email into the incoming queue with the help of the clean-up program. This clean-up program signals the arrival of new mail to the queue_manager program, checks the headers (as for instance the hostname of the sender), eventually informs the administrator and rewrites the messages in the incoming queue. All emails in this queue are transferred to the queue manager which is the core of the postfix system. After having given the necessary parameters as the name and address of the recipient, the queue manager puts the message into an active state (if the email is not temporary deffered) and thus calls an appropriate delivery agent (as local, bounce, rewrite or smtp) . Local messages (from and to our domain) are transferred on the server which hosts the mailboxes *via* the local agent. Emails from our client Workstations (*i.e.* by a "Mail User Agent", as for instance mailx , outlook, eudora, pine, etc.) are sent through the Internet using the SMTP utility. External mails from the Internet are received by the SMTPd daemon and written into the incoming queue with the help of the clean-up program and, then, any external emails follow the same process as a local message.

A MailScanner child process picks up a batch of messages from the incoming mail queue and first runs its own Real-time Black List (RBL see below) checks on the messages in the batch.
SpamAssassin (`http://www.apache.org`) is probably the only open-source software able to determine the probability for a message to stand for a spam. Furthermore, MailScanner works solely with SpamAssassin[4].

If MailScanner is configured to use SpamAssassin, it then calls SpamAssassin once for each batch of messages (not once for each message), by directly calling the SpamAssassin Perl modules, not the executable spamassassin or spamd, and runs the SpamAssassin rules against this batch of messages.

[4] SpamAssassin is a spam-scoring engine used by many commercial products [10].

Real-time Black List Many black lists, which compile the IP address of the spam servers, exist over the world. These lists are renewed constantly and they offer a first level of protection against spam.

MailScanner runs a series of RBL tests on each message. If the IP address of the sender's mail server or mail relay servers matches one of the addresses on the lists, the message may marked definitively as spam and no further tests are performed [11].

Those messages which pass the RBL tests, are passed to SpamAssassin which use heuristic, Bayesian and other tests to determine the spam level of the message (*i.e.* a spam "score").
SpamAssassin and its Bayesian filtering SpamAssassin engine is present in most of the commercial spam scanners. SpamAssassin is also an Open Source software, recognized as the most intelligent tool to prevent the propagation of spam.

SpamAssassin assigns a numerical value to each test that is used on the message. SpamAssassin also examines the site specific white lists (ham *i.e.* not spam) and black lists (spam). SpamAssassin calculates the final score for each message at the end of these tests.
The Bayesian filter in SpamAssassin is one of the most effective techniques for filtering spam.

Although Bayesian statistical analysis is a branch of mathematics, one doesn't necessarily need to understand the mathematics to use spamassassin's Bayesian filter. Bayesian analysis involves teaching a system that a particular input gives a particular result. For spam filtering, this teaching is repeated, many times over, with many spam and ham emails. Once this is finished, a Bayesian system can be presented with a new email and will give a probability of the result being spam. For best results, teaching should be a constant process.

Internally, the Bayesian engine provides a single probability figure for each email processed. This probability ranges from 0 (0% likelihood that an email is spam) up to 99 (99% likelihood) [8].

To filter spam emails, the system is taught both ham and spam emails, until the filter has learned to differentiate between the two. Then, emails passed through the filter will be assigned a probability of being spam. When Bayesian filtering is used in conjunction with SpamAssassin's other spam detection rules, SpamAssassin approaches 100% detection of spam, with false positives (legitimate emails misclassified as spam) close to 0%.

After this stage of the processing, MailScanner has all the information needed to modify, deliver, reject or quarantine the message. Once SpamAssassin has assigned a numerical value to the messages, MailScanner can perform any combination of the following configurable options (see fig. 1):

- Delete - delete the message
- Store - store the message in the quarantine
- Bounce - send a rejection message back to the sender
- Forward - forward a copy of the message to user@domain.com

– Strip HTML - convert all in-line HTML content to plain text
– Attachment - Convert the original message into an attachment of the message
– Deliver - deliver the message as normal

3.4 Treatment of viruses

The principal vector of viruses is email. To run a "good" virus scanner on an email gateway is thus the best protection. Indeed, some worms propagate through emails. Once installed on a computer, this kind of virus uses the addresses book of the recipient to infect his or her contacts.

Email worms are not systematically recognized by the clients virus scanner. MailScanner may be configured to use one or more of seventeen commercial or Open Source virus scanners. If a virus is detected, the message is marked as containing a virus in the subject. We have performed several tests to choose the most appropriate bundle of virus scanners, aggregate to MailScanner.
The criteria used are:

– The reactivity of the virus definition
– The time taken in treating a message
– The cost (price by server vs. by mailbox),
– The combined use of Open Source and commercial virus scanners

The virus scanners tested in our study were *Antivir*, *bitDefender*, *clamAV*, *FProt*, *F-Secure*, *McAfee*, *Sophos*, *Command* and *Kaspersky.* In our case, a server solution is more advantageous than a mailbox analysis, so we finally decided to use clamAV (`http://www.clamav.net`), Kaspersky (`http://www.kaspersky.com`) and Command (`http://www.authentium.com`).

Once virus detection is complete, the MailScanner child process examines the filename and file types of any email attachments against configurable rule sets. We note that this software can perform tests on zip archives that are not password protected (with as many as 7 levels of compression).

3.5 Treatment of malicious attachments and other attacks

The message is also examined by MailScanner to see if the body contains possibly dangerous HTML content such as `IFrame` or `<Form>` tags. Configurable options allow logging, passing, deleting or disarming these HTML content tags.

Over the past two years, we have seen the proliferation of a new type of attack. This technique, called *phishing*, also referred to as brand spoofing or carding, has already caused a lot of damage. Messages are sent to a user purporting to be from an established legitimate enterprise in an attempt to elicit the user into surrendering private information that will be used for identity theft. The email directs the user to visit a Web site where they are asked to update personal information, such as passwords and credit card, social security, and bank account numbers, that the legitimate organization already has. The Web site, however,

is bogus and set up only to steal the users information.
A new module incorporated in MailScanner allows detection of such a message.

If a virus or another dangerous tag is detected, MailScanner can send:

- A customized message to the sender of the virus (normally not desirable)
- A customized message to the recipient of the virus
- The disarmed and sanitized message to the recipient
- The message and the virus to quarantine
- The disinfected or cleaned message to the recipient

3.6 Front-end related applications

Other applications may be installed with MailScanner to simplify administration and provide additional functionality. We describe only here MailWatch, which is a web-based front-end to MailScanner written in PHP, MySQL and JpGraph and is also freely available under the terms of the GNU Public Licence.

MailWatch (`http://mailwatch.sourceforge.net`) contains a module which causes MailScanner to log all message data (excluding body text) to a MySQL database. MailWatch is then able to display reports and statistics as follows [11]:

- Load average and daily totals for messages, spam, viruses and block content on each page header
- Reports with customizable filters and graphs (see fig 3)
- Color-coded recently processed emails
- Drill-down onto each message to see detailed information
- Quarantine management and spam learning
- MySQL database status
- MailScanner configuration files

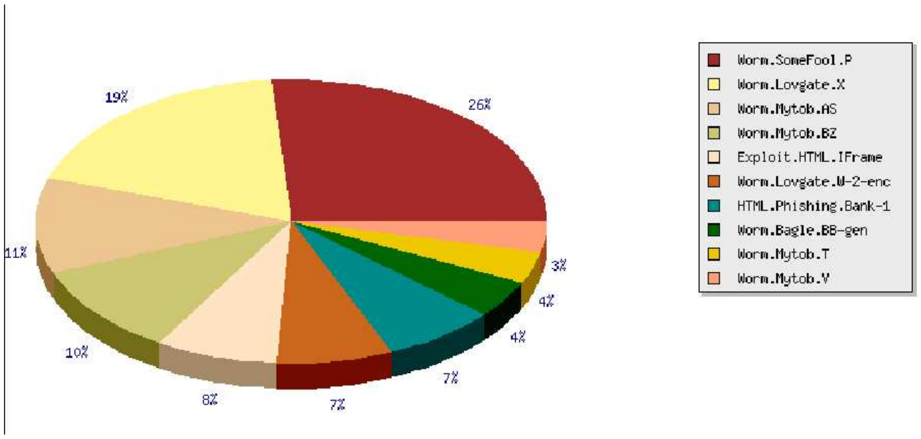

Fig. 3. Example of reports displayed by MailWatch for MailScanner: top ten viruses received by our SMTP server are shown in a pie chart.

3.7 Tuning and optimization

DNS cache To improve time of RBLs testing, installation of a DNS cache server is highly recommended. Then, after a short while, the IP addresses compiled on these lists are known by the server and no other DNS request is needed.
BIND is an Open Source secure DNS server, adapted to this use (`http://www.isc.org/`). Implementation of BIND on a Linux box is quite easy and well documented.

tmpfs file system MailScanner "unpacks" messages for scanning on the directory `/var/spool/MailScanner/incoming`. Mounting this directory in memory improves dramatically performance. Of course, no email will be lost even if the system crashes. The software never removes a message from the incoming mail queue until it is fully written to the outgoing mail queue. In case of system crash, when MailScanner restarts, it will find the "lost" emails in the incoming mail queue and it will process these messages normally.

Modification of the logging facility In a normal use, the syslog daemon notices the system in a very explicit (verbose) manner, especially when MailScanner starts. Speed logging can be obtained by a simple modification of the `/etc/syslog.conf` file in a way to omit the synchronising of logs.
Improved performance is obtained using this method.

Confidence in our domain As the spam checks takes the most time, Spamassassin can be configured to ignore mail within the Local Area Network (LAN). It presupposes that no spammer connects from within the LAN but this method is probably the most effective way to reduce the internal delivery time. For this, one can simply modify `/etc/mail/spamassassin/local.cf`.

4 Conclusions and Perspectives

As underlined in the introduction, communicating by email has become crucial in the day to day work of sites such as government departments, commercial businesses and educational institutions. Nevertheless unwanted and unsolicited communications are responsible for a significant loss of time and money for everybody concerned and may be damaging for corporate reputations.

It appears from this study that it is possible to implement a secure SMTP gateway, in a working context, with only Open Source software. Such an email gateway is a tool able to stop viruses, to filter spam, to detect attacks against security vulnerabilities and thus plays a major part in the security of a network. The financial cost associated with this gateway is null, except for the investment in the two servers. A schematic view of this gateway is shown in fig. 4.

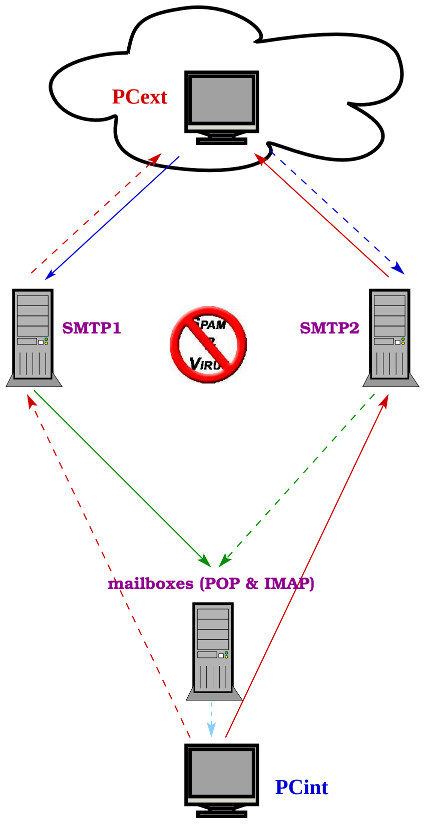

Fig. 4. Scheme of the distribution of emails from/to outside (PCext) from/to inside our network. Red arrows show the outbound SMTP fluxes, blue ones represent the incoming fluxes. Green arrows represent the transport of messages to the server containing the mailboxes by LMTP (Local Mail Transfer Protocol). Light blue arrow illustrates a request (POP or IMAP) by a client. SMTP1 and SMTP2 are the same and load balanced: if a server crashe, the second one will take its place. Both host running virus scanners and spam scoring software are represented in this figure.

The low-cost secure email gateway described here has been deployed for one year at the INT. On the approximately 60 million messages received per year in our institute, we can note that only 25% contains real information: around 63% of spam and 9% of viruses[5]. Today, spam filtering is more than 94% efficient and it is also important to notice that one false positive appears once in a month. Most of the spams which are not identified are due to a fast mode of MailScanner used to speed up the delivery process. All viruses are stopped by this gateway and

[5] Theses rates include internal emails.

most of the attacks are detected. However, we arbitrary stop many attachments, as the Windows executable files, which somehow disturb users during the first two months following the installation of this gateway. Now, users agree in saying that this gateway provides them with an improved working environment.

The next step will be now to guarantee the complete availability of the mailboxes by keeping the same principle, *i.e.* to distribute the load across multiple machines rather than buying bigger and faster machines.
Instead of using the SMTP protocol to send messages, clients use POP or IMAP to retrieve messages. The Post Office Protocol version 3 (POP3, specified in RFC 1734) and the Internet Message Access Protocol (IMAP, described in RFC 1731) are the two main email retrieval protocols, used [9]. The underlying application we use is Cyrus-imapd which allows the mailboxes to be POP compliant with a full support of IMAP (documentation about Cyrus IMAP server can be found at `http://asg.web.cmu.edu/cyrus/imapd/`).
This means that users can access the mail from virtually any email client, and choose to have it in the usual POP "stored-forward" method, which downloads to their client, or IMAP's storage method, retaining the mail on the server.

This hosting mailbox server is clearly a single point of failure. The approach of distributing the load among IMAP/POP servers generally sacrifices the unified system image (distributing the load is complicated, particularly since there is no concept of mailbox location in IMAP). For pure email, this is an acceptable compromise; however, trying to share mailboxes becomes difficult or even impossible. A new approach to overcome these problems exists and is called Cyrus IMAP Aggregator. The Cyrus IMAP Aggregator transparently distributes IMAP and POP mailboxes across multiple servers, thereby appearing to be only one server to the clients. Unlike other systems for load balancing IMAP mailboxes, the aggregator allows users to access mailboxes on any of the IMAP servers in the system. Insights on the way to install and to configure Cyrus IMAP Aggregator can be found at `http://asg.web.cmu.edu/cyrus/ag.html`.

Acknowledgements

Author thanks Nigel Barnett for his help with the translation of this paper. I acknowledge all the members of S2IA for many helpful discussions. I thank Anas Abou El Kalam for his concern in the submission of this article.

References

1. Aoun, F., B. Rasle, B.: *Halte au Spam*, (Eyrolles, 2003).
2. Dent, K. D.: *Postfix Definitive Guide*, (O'Reilly, 2004).
3. *Calculating Spam Cost for your Organization*, **511** Ferris Research, 2005.
4. Field, J.: *MailScanner: A User Guide And Training Manual*, EPrint Type Book, 2004.
5. Graham, P.: *Adaptive Filtering: One Year On*, in Usenix LISA, **03**, 2003.

6. Graham, P.: *A Plan for Spam*, (web publication; IETF mailing list discussion), 2002.
7. Licklider, J.C.R.,Taylor, R.W.: *The Computer as a Communication Device.* In Science and Technology: For the Technical Men in Management, **76** (1968) 21–31.
8. McDonald, A.: *SpamAssassin: A practical guide to integration and configuration* 73–80 (Packt Publishing, 2004).
9. Mullet, D., Mullet, K.: *Managing IMAP*, (O'Reilly, 2000).
10. Schauer, H.: *Relais de messagerie sécurisé et libre*, SETI proceeding, CS2, 2004.
11. Swaney, S., Bellavance, U., Neylon, M.: *MailScanner administrator guide* (Fortress Systems Ltd., 2004).
12. Williamson, A.: *LWM Speaks with Julian Field: MailScanner addresses the growing spam problem*, in LinuxWorld Magazine, December 2003.