Filtrage de messagerie sur des gros serveurs

José-Marcio Martins da Cruz
Ecole des Mines de Paris – Centre de Calcul
60, bd Saint Michel
75275 – Paris
Jose-Marcio.Martins@ensmp.fr

Résumé

Actuellement, les performances des filtres anti-spam sont plus critiques que la capacité de routage des serveurs SMTP. Ceci est un domaine récent et plusieurs solutions sont disponibles. Les résultats que l'on obtient avec ces solutions sont très bonnes, voire excellentes, lorsqu'il s'agit du filtrage adapté à un seul utilisateur ou alors sur des serveurs petits ou moyens : jusqu'à quelques dizaines de milliers de messages par jour vers quelques centaines d'utilisateurs. Mais lorsqu'il s'agit de filtrer des centaines de milliers de messages par jour pour un ensemble de plusieurs milliers d'utilisateurs, d'autres problèmes apparaissent et l'efficacité de ces solutions diminue.

Ce papier présente les points qui font que les méthodes de filtrage applicables à des petites communautés donnent des résultats moins bons quand ils sont utilisés sur des serveurs de taille importante. Ces points expliquent les choix effectués lors du développement de j-chkmail.

Mots clefs

Serveurs de messagerie, MTA, sendmail X

1 Introduction

L'Internet est devenu un outil de communication grand public il n'y a pas beaucoup plus de 10 ans. C'est un outil qui a modifié profondément la société partout dans le monde.

En 1988 apparaissait le premier ver se propageant par la messagerie [1] mais s'attaquant exclusivement aux machines sous UNIX. Les virus existaient déjà à l'époque, mais ils se propageaient autrement (par exemple, transfert de fichiers bureautiques).

Jusqu'à 1999 l'activité des virus de messagerie est restée marginale : la protection des postes des utilisateurs était suffisante. *Melissa* [2] a été le premier virus de messagerie à propagation rapide (pour l'époque) mais, étant embarqué dans une macro (script) Microsoft Word, il nécessitait la présence de ce logiciel sur le poste du destinataire pour pouvoir l'infecter.

Love Letter [3] est apparu, peu de temps après, exploitant une vulnérabilité dans la gestion des scripts par le système d'exploitation, et utilisé par le client de messagerie. Il suffisait d'ouvrir (double cliquer) le fichier attaché pour s'infecter, et pour que le virus puisse être retransmis à tous les contacts présents dans le carnet d'adresses de l'utilisateur. A partir de ce moment, le filtrage viral sur les serveurs de messagerie est devenu pratiquement indispensable.

L'activité *spam*[1] a toujours existé sur Internet, mais elle est restée marginale pendant longtemps. Vers 1997 l'utilisation massive des relais ouverts est devenue problématique, mais pouvait être résolue avec une configuration correcte des serveurs de messagerie. Cette activité a augmenté de plus en plus rapidement jusqu'à ce qu'elle devienne ce qu'elle est aujourd'hui.

La partie du trafic de messagerie correspondant au *spam* et aux virus est actuellement comparable ou même supérieure à celle du trafic légitime. A tel point que ce sont maintenant ces points qui déterminent l'architecture du système de messagerie des organisations [4].

Au niveau d'un serveur de messagerie, il s'agit de choisir le type de filtrage capable donner des bons résultats, non seulement en efficacité, mais aussi en performance et consommation de ressources. Ce dernier point est particulièrement critique car le volume à traiter est important.

Des résultats excellents peuvent être atteints avec les filtres *anti-spam* « open source » disponibles : Bogofilter [5], CRM114 [6], DSPAM [7], Spamassassin [8], etc. Néanmoins, la plupart d'entre eux privilégie les résultats du filtrage lui-même plutôt que les performances ou bien ils ne s'appliquent efficacement qu'à des populations homogènes et aucun n'inclut des mécanismes de protection du serveur.

Le développement de j-chkmail a commencé début 2002 comme une solution provisoire au filtrage de virus sur un serveur de messagerie, basé sur la détection des messages ayant des fichiers attachés exécutables. L'idée de départ était le développement d'un outil peu gourmand capable de traiter le trafic d'une organisation d'un millier d'utilisateurs tout en utilisant un ordinateur de taille modeste.

Le filtre a évolué avec l'ambition d'être utilisable aussi sur des serveurs de taille importante. Le principe initial d'économie de ressources est indispensable pour l'utilisation sur des gros serveurs, mais ce n'est pas le seul. Il faut aussi tenir compte des limitations liées à un nombre important d'utilisateurs et aux aspects sécuritaires qui n'existent presque pas sur les petits serveurs.

Ce papier rappelle les caractéristiques des « gros serveurs » et les caractéristiques générales des méthodes de filtrages (des points peut-être redondants, mais qui d'habitude ne sont pas suffisamment présentés). Ensuite, on présente les principes et fonctionnalités implémentées dans j-chkmail, suivies des idées en cours d'étude dans un proche avenir.

[1] Spam – mot utilisé pour référencer un message non sollicité

2 Les gros serveurs de messagerie

Certaines caractéristiques des serveurs de messagerie de taille importante font qu'on ne peut y transposer les résultats obtenus lors des expérimentations sur des petits serveurs. Il peut y avoir des raisons liées à la diversité des destinataires, au niveau de trafic ou alors aux limitations du matériel et du système d'exploitation :

- **Diversité des utilisateurs** – les profils des utilisateurs dans un campus universitaire peuvent être très variés : des sociologues, des administrateurs, des économistes, des informaticiens, des médecins, etc. Cette diversité fait que la description, aussi bien qualitative que quantitative, que l'on peut faire de la BAL[1] d'un utilisateur (ou d'un groupe d'utilisateurs) n'est pas applicable à la communauté entière et vice versa.
- **Facteurs humains** – les rapports humains entre les administrateurs de la messagerie et les utilisateurs dans une petite organisation sont beaucoup plus simples que dans une organisation importante, où parfois ils ne se connaissent même pas. Il est souvent difficile d'obtenir des informations de retour fiables permettant de mettre au point les filtres, dans ce cas il arrive que, pour éviter les situations de conflit, des solutions plus efficaces ne puissent pas être mises en place.

  Les situations de conflit sont aussi plus fréquentes à cause des messages qui n'arrivent pas à destination, ou avec des délais supposés inacceptables. Il est courant d'être confronté à des utilisateurs agacés qui s'en prennent aux administrateurs du serveur de messagerie le plus proche.
- **Extensibilité** – il n'est pas trop compliqué d'avoir un système de filtrage capable de traiter 20000 messages par jour pour 1000 utilisateurs. Néanmoins, si l'on souhaite traiter un trafic dix fois plus important, un critère à rechercher est que la puissance de calcul nécessaire doit croître moins rapidement que la quantité de trafic à traiter.
- **Fiabilité / Disponibilité** – sur un serveur de grande taille, les arrêts de fonctionnement et les dysfonctionnements sont moins bien tolérés que dans les petits serveurs.
- **Architecture du service de messagerie** – Dans des organisations de taille importante, le service de messagerie est organisé de façon à ce que le serveur en entrée du site ne fasse office que de passerelle, les messages étant stockés sur des serveurs plus proches de l'utilisateur. Dans ce cas, les informations sur les utilisateurs ne sont pas toujours disponibles sur la passerelle.
- **Évènements inattendus** – un serveur de messagerie de grande taille doit être capable de traiter correctement ou d'avoir le comportement adéquat face à des charges importantes et passagères, qu'il s'agisse des attaques ou du trafic normal, et en aucun cas s'écrouler dans ce type de situation.
- **Les goulots d'étranglement** – sur des serveurs petits et moyens, on néglige souvent, à juste titre, certaines limitations de bas niveau qui vont apparaître lorsqu'on traite un niveau de trafic important : la place mémoire occupée par le processus de filtrage, la bande passante des entrées/sorties (disque ou réseau), le nombre de processus présents sur la machine, les méthodes d'accès optimales aux informations en mémoire ou sur disque, etc.

[1]BAL – Boîte aux Lettres

3 Le filtrage statistique

Le contenu de cette section a pour but la présentation, même sommaire, des idées théoriques communes à tout filtre anti-spam statistique, de comprendre pourquoi il y a des erreurs et pourquoi les résultats obtenus dans des environnements limités ne peuvent pas être extrapolés à des gros serveurs avec des communautés importantes.

Du point de vue de l'utilisateur final, le but recherché serait de ne pas recevoir un seul *spam* (taux de faux négatifs nul) et ne pas avoir un seul *ham*[2] bloqué par erreur (taux de faux positifs nul).

Il s'agit, donc, d'un processus de classification, dans lequel on va utiliser les caractéristiques connues des *hams* et des *spams*, pour décider dans quel groupe classer un nouveau message arrivant. Un processus de classification est, par nature, un processus statistique.

Il y a quatre cas de figure de décision que l'on peut prendre :

Message \ Décision	***ham***	***spam***
ham	Correct	Faux Positif
spam	Faux Négatif	Correct

Table 1 – *Bonnes et mauvaises décisions de filtrage*

Le premier point à régler est le choix du critère de classement. Tous les critères n'ont pas la même efficacité.

Pour comprendre, prenons un exemple simple (peu efficace) : la présence du mot « *viagra* » dans le texte du message. On dira qu'il s'agit d'un *spam* si ce mot apparaît dans le message et d'un *ham* en cas contraire. Mais si ce mot peut naturellement apparaître dans un message légitime, il y a beaucoup de *spams* dans lequels ce mot n'apparaît pas. Il est possible de construire des critères de plus en plus complexes, mais des exceptions pourront toujours exister.

Généralement, le processus de filtrage consiste d'abord à analyser l'objet d'étude avec le critère choisi et de lui attribuer un score (mesure) selon qu'il remplit plus ou moins ce critère. Ce score est, en réalité, une mesure de la distance séparant l'objet des deux classes.

3.1 Une analogie avec les systèmes de communication...

Pour pouvoir clarifier les limitations des filtres anti-spam statistiques, il est intéressant de regarder le processus de filtrage comme une analogie avec les processus de détection et identification que l'on étudie dans les systèmes de communication [9].

[2]Ham – message légitime, par opposition à spam.

La situation est alors décrite comme une (une seule) source qui émet une information avec deux valeurs possibles (disons un *spam* ou un *ham*). Le canal de communication ajoute un bruit aléatoire additif caractérisé par une certaine fonction de distribution de probabilité. Ce bruit fait que le *spam* peut ressembler à un *ham* (ou vice-versa). Le problème du récepteur est de classer l'information reçue et inférer l'information effectivement envoyée.

La représentation graphique de cette situation permet de la comprendre plus facilement et d'identifier l'analogie avec les méthodes de filtrage de *spam* usuels.

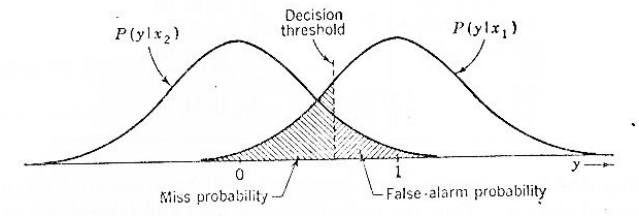

Figure 1 – *Probabilité de fausse alerte et de perte en fonction du seuil de décision* ([9] – page 222)

Les courbes en cloche de gauche et de droite correspondent, respectivement, à la distribution statistique des scores des *hams* et des *spams*, évalués selon les critères de filtrage. Ces courbes sont constituées à partir d'un corpus de messages d'un utilisateur, et représentent le flux de messages arrivant. Pour chaque courbe en cloche, la partie gauche correspond aux critères apparaissant prioritairement dans les *hams* et la partie droite aux critères apparaissant prioritairement dans les *spams*.

La probabilité de fausse alerte (faux positif) est la portion de la courbe de gauche qui se trouve à droite du seuil de décision, ainsi que la probabilité de perte (faux négatif) est la partie de la courbe de droite qui se trouve à gauche du seuil. Ces probabilités d'erreur sont données par les relations :

$$\begin{aligned} P[FN] &= P[d=ham \mid m=spam] \,.\, P[spam] \\ P[FP] &= P[d=spam \mid m=ham] \,.\, P[ham] \\ P[Error] &= P[FN] + P[FP] \end{aligned} \quad (1)$$

Les probabilités absolues, P(*ham*) et P(*spam*), représentent la proportion de *ham* et de *spam* dans un corpus de messages reçus par un utilisateur, alors que les probabilités conditionnelles représentent, pour un seuil donné, les probabilités d'erreurs, compte tenu des critères de filtrage identifiés – ce sont les surfaces hachurées dans la figure ci-dessus.

La possession de ces informations statistiques permet de choisir un seuil de détection minimisant une des trois probabilités d'erreur.

Des courbes de la figure 1, on peut déduire :

- Si l'on augmente de seuil de décision, la probabilité de fausse alerte (faux positif) diminue, tandis que la probabilité de perte (faux négatif) augmente.
- Moins les courbes se chevauchent, plus faibles sont les probabilités d'erreur.
- Plus la courbe des scores des *hams* est abrupte à droite et la courbe des scores des *spams* est abrupte à gauche,, moins elles se chevauchent.
- Si la courbe de gauche est très ouverte à droite, cela signifie que l'on trouve, dans les messages légitimes, des caractéristiques qui peuvent faire croire qu'il s'agit d'un *spam*, comme par exemple, message en HTML riche, des mots douteux (viagra, ...).
- Si la courbe de droite est très ouverte à gauche, cela signifie que l'on trouve, dans les *spams,* des caractéristiques qui peuvent faire croire qu'il s'agit d'un message légitime, comme par exemple, un texte sobre avec beaucoup de mots « savants ».
- Les expéditeurs ont intérêt à générer des messages dont la courbe de distribution est abrupte à droite (pas trop de critères noircisseurs) – dans ce cas, les filtres peuvent avoir un seuil plus faible avec un petit taux de faux positifs. Mais ils souhaitent ne pas avoir ce souci, ils estiment que c'est aux filtres s'en occuper. Un message légitime au format HTML avec des couleurs et images de fond, du texte avec plusieurs polices, tailles et couleurs, risque d'être confondu avec un *spam*.
- Les spammeurs ont intérêt à générer des messages dont leur distribution est ouverte à gauche (utiliser des critères blanchissants). Dans ce cas, ils obligent les filtres à utiliser un seuil important de façon à que le taux de faux positifs soit acceptable.
- Les administrateurs des filtres ont intérêt à supprimer certains critères de filtrage de façon à que les courbes en cloche soient optimales : la courbe de *spam* abrupte à gauche (sans critère blanchissant) et la courbe de *ham* abrupte à droite (pas de critère noircissant).

Ces raisons expliquent pourquoi, même en théorie, les filtres anti-spam ne sont pas parfaits, et cela même pour un utilisateur seul. Dans la pratique, d'autres facteurs viennent s'ajouter.

On doit se poser la question de la faisabilité de la modélisation de la BAL typique d'un utilisateur, et en particulier de l'identification des critères pertinents qui permettront le classement *ham/spam*.

En première approximation, on peut néanmoins émettre l'hypothèse que les caractéristiques des *hams* sont constantes.

Par contre, les caractéristiques des *spams* changent continuellement. Pour résumer, du point de vue du spammeur, un générateur de messages optimal est celui qui :

- génère des messages avec des caractéristiques propres aux *hams* (courbe avec variance importante) pour forcer les filtres à utiliser des seuils importants et détecter peu de *spams*.
- n'est pas stationnaire : ses caractéristiques changent avec le temps pour rendre complexe la mise au point des filtres et obligatoires les mises à jour périodique ou, à défaut, les méthodes d'apprentissage, permettant d'intégrer l'évolution temporelle des caractéristiques des *spams*.

De ces idées ressort que tout filtrage statistique genère une probabilité d'erreur non nulle et souvent non négligeable. Toutefois, il y a un et un seul filtre capable d'atteindre une efficacité optimale : le « filtre humain ». L'utilisateur lui-même est le seul à pouvoir modéliser correctement sa BAL, mais aussi à utiliser des algorithmes bien plus complexes que ceux implémentés dans les filtres courants.

Mais l'expérience montre que si l'être humain est capable de trier correctement sa BAL, il ne l'est (généralement) pas pour celle des autres.

3.2 Un point de vue statistique

Nous avons montré graphiquement que l'on peut obtenir des résultats de filtrage meilleurs lorsque les « courbes en cloche sont abruptes ». Or, le paramètre statistique qui caractérise la dispersion des valeurs d'une distribution de probabilités est la variance.

L'inégalité de Chebischev [10] ne donne qu'une borne supérieure dans une prise de décision de filtrage, mais elle explique le comportement d'un filtre et la confiance que l'on peut attribuer à un changement d'environnement :

$$P[(X-\mu) \geq t^2] \leq \frac{\sigma^2}{t^2} \quad (2)$$

Ainsi, considérons le score des *hams* de la boite aux lettres d'un utilisateur, caractérisé par une distribution de probabilité de moyenne μ et variance σ^2 . Avec les mêmes critères de filtrage et le même seuil, si l'on prend une deuxième BAL dont la distribution du score est caractérisée par la même moyenne, mais dont la variance est le double, on peut espérer que le taux de faux positifs sera le double. On arrive à la même conclusion, au sujet du taux de faux négatifs, si l'on considère les scores des *spams*.

Dans une organisation de taille importante, une des caractéristiques de la messagerie est la diversité des BALs des utilisateurs : en ce qui concerne la proportion *ham/spam*, les langues, la forme, les sujets, etc.

La question qui se pose est : comment caractériser la BAL globale à partir des boîtes individuelles ? L'affirmation intéressante que l'on peut faire est que lorsqu'on regroupe deux BALs, la variance est au moins égale à la plus grande des deux.

La conclusion que l'on peut tirer est que lorsqu'on regroupe des BALs de caractéristiques différentes, la variance augmente et, par conséquent, si l'on utilise les mêmes critères de filtrage, les erreurs augmentent et l'efficacité du filtrage diminue.

3.3 Deux filtres en cascade

Ce cas d'utilisation est intéressant puisqu'il montre que, même si les résultats de filtrage semblent bons, la perception que l'on peut avoir change selon les caractéristiques du flot à traiter. Ce cas montre aussi que les indicateurs d'efficacité n'ont un sens que si les conditions de mesure sont précisées.

Supposons, par exemple, le cas typique d'un flot dont la moitié est constitué de *spams*, avec un filtre heuristique qui détecte 85 % des *spams* avec un taux de faux positifs de 1 %. Si l'on insère un filtre basé sur greylisting, avant le filtre heuristique, capable de supprimer 90 % du *spam*, l'efficacité du filtre heuristique ne sera pas la même et peut être prévue numériquement par l'application de relations (1)., que nous résumons dans la Table 2.

Il a suffit de changer le rapport *spam/ham* pour que les valeurs de probabilité d'erreur deviennent complètement différentes.

Ce cas est assez trompeur (typique des problèmes avec des probabilités conditionnelles que l'on voit à l'école) et pas toujours évident à expliquer à un utilisateur final.

	Avant	***Après***
Messages arrivant	100	55
Rapport *spam / ham*	0,50	0,09
Faux positifs	1 %	1,8 %
Faux négatifs	15 %	2,7 %

Table 2 – *Changement de l'efficacité d'un filtre heuristique après avoir inséré un filtre basé sur greylisting avant lui*

Cet exemple considère vraie l'hypothèse que l'action du premier filtre est distribuée uniformément sur toutes les catégories de spam – ce qui est sûrement faux. Le greylisting, par exemple, est une méthode de filtrage dont le critère n'a aucun lien avec les scores obtenus par un filtrage de contenu et, par conséquent, n'aura aucun effet sur les *spams* envoyés à partir des serveurs de messagerie normaux ou des webmail (ce qui est souvent le cas pour les *scams*[1] africains). Ainsi, on ne peut rien dire sur la forme de la courbe de distribution des scores en entrée et en sortie d'un filtre greylisting, mais il est peu probable qu'elles restent identiques.

4 Le filtrage (presque) déterministe

Comme nous l'avons vu, le filtrage sans erreurs, ni faux positifs ni faux négatifs, est impossible avec les techniques disponibles actuellement.

Néanmoins, il existe des filtres qui peuvent avoir soit la probabilité de faux positifs ou de faux négatifs (l'une ou l'autre, mais pas les deux) nulle ou suffisamment basse pour qu'elle puisse être négligée. Pour cette raison, on les appelle « déterministe ».

Ses décisions sont binaires, mais une seule parmi les réponses est sure.

L'efficacité globale de ces filtres est sûrement limitée, mais ils sont utiles lorsqu'on veut s'assurer que le résultat du filtrage appliqué à une catégorie de messages sera déterministe, donc prévisible.

4.1 Les filtres *blancs*

Ces filtres se contentent de classer les messages en deux catégories : *hams* et « le reste », cette dernière classe contient aussi bien des *hams* et des *spams*. Un exemple de critère de filtrage de ce type est l'utilisation de DKIM associé à une liste blanche. On peut, par exemple, établir

[1]Scam – mot anglais - escroquerie

que les messages signés avec un certificat et émis par le CRU peuvent passer sans subir aucun autre filtrage.

4.2 Les filtres *noirs*

C'est le dual des filtres *blancs*. Il sert à détecter, sans erreur, une catégorie de *spams*.

La liste noire d'URLs gérée par surbl.org [11] est une liste noire d'URLs apparaissant dans les *spams*. La règle utilisée par les gestionnaires est « If it appears in *hams*, don't list it ». Cette règle fait que, au moins dans ses objectifs, elle peut être classée comme étant une méthode de filtrage déterministe. En effet, son taux de détection est de l'ordre de 70 à 80 %, avec un taux de faux positifs inférieur à 0,1 %.

mail-abuse.com [12] est une liste noire d'adresses IP, dont les observations faites par l'auteur montrent que de l'ordre de 97 % des connexions bloquées proviennent des adresses IP résidentielles. Certaines organisations peuvent estimer que les messages en provenance des adresses résidentiels ne les intéressent pas. Dans ce cas, l'utilisation de cette méthode de filtrage peut être considérée comme déterministe.

L'intérêt de ce deuxième exemple est de montrer que des critères de filtrage peuvent ne pas avoir le même intérêt selon l'organisation.

5 Filtrage comportemental versus filtrage de contenu

La classification des filtres en statistiques ou (presque) déterministes n'est pas la seule possible. On peut aussi classer une méthode de filtrage en comportemental ou de contenu.

Le filtrage de contenu se base uniquement sur le contenu d'une seule connexion SMTP ou message. Le filtrage de contenu peut être fait sur un critère simple, comme par exemple l'existence du mot *viagra*, mais peut aussi être basé sur des analyses textuelles bien plus complexes donnant des résultats bien plus précis, comme c'est le cas, par exemple de CRM114, qui annonce une probabilité d'erreur de classification inférieure à 1 % mais dont la vitesse de traitement n'est que de 20 Koctets par seconde sur un Intel Pentium 666 Mhz [6]. Malgré les excellents résultats quantitatifs de filtrage, ces filtres ne sont pas suffisamment rapides pour être utilisés sur un gros serveur de messagerie.

Contrairement au filtrage de contenu, le filtrage comportemental cherche à repérer des « mauvais » comportements répétés associés à une même adresse IP. Ces comportements à identifier peuvent être soit des caractéristiques liées aux connexions (rafales de connexions, trop de connexions ouvertes en même temps) ou encore certaines caractéristiques du contenu comme, par exemple, des messages consécutifs vers des utilisateurs inconnus, ou des *spams* consécutifs (vu que les cinq derniers messages venant d'un certain client étaient des spams, il est fort probable que le sixième le soit aussi).

L'inconvénient du filtrage comportemental est le besoin de gestion d'un historique (en mémoire ou disque), dont la longueur dépend du comportement que l'on souhaite observer.

Mais ce qui peut devenir intéressant est la coopération entre des filtres de contenu et des filtres comportementaux. Ceci permet, en principe, l'utilisation de méthodes de filtrage de contenu plus complexes, le résultat étant utilisé comme un « comportement » à vérifier. Selon la constance de ce comportement, à vérifier sur une fenêtre temporelle, les traitements suivants peuvent être plus légers.

La coopération entre le filtrage comportemental et de contenu peut exister dans les deux directions, mais des précautions doivent être prises pour éviter des boucles, ce qui pourrait générer des instabilités et des erreurs de filtrage.

6 Traiter un trafic important

L'extensibilité est une mesure de l'accroissement dans la consommation de ressources, par le filtre, lorsqu'on augmente le niveau de trafic à traiter. Idéalement, la consommation de ressources doit croître moins rapidement (pas plus que linéairement) avec le niveau de trafic.

Cette section présente les idées implémentées par j-chkmail dans le but de le rendre extensible.

Filtrage comportemental – L'idée est d'utiliser la corrélation qui peut exister entre les connexions/messages successives venant du même client SMTP. On peut comparer ceci à une liste noire constituée au fur et à mesure du fonctionnement du filtre : des clients qui présentent un comportement mauvais et répété ou dangereux se trouvent bloqués pour une durée plus ou moins longue, selon le paramètre utilisé pour évaluer son comportement. Ceci permet de bloquer des messages sans évaluer le contenu des messages.

Privilégier les critères objectifs – ce sont des critères dont le taux de faux positifs est suffisamment bas. Pour ces critères, l'on peut appliquer des mesures curatives [4]. Pour les critères dont les résultats ne sont pas suffisamment sûrs ou dépendants des utilisateurs, on va se contenter d'attribuer un score qui sera ajouté dans un en-tête.

Prendre des décisions dès que possible – D'une part les traitements effectués lors des phases préliminaires du protocole SMTP sont bien plus légers que le traitement du contenu, et d'autre part, à cause des constantes de temps plus importantes, les dépendances sont plus fortes dans la phase DATA. Donc, si dès le traitement de l'enveloppe on sait déjà que le message sera refusé, il est inutile d'attendre l'analyse de contenu pour le faire.

Éviter les traitements longs – Tout traitement long est à éviter, qu'ils soient consommateurs de puissance CPU ou constitués uniquement de temps d'attente. Les traitements avec des temps d'attente font croître le nombre moyen de processus sur la machine.

Proscrire les traitements dont la complexité algorithmique est importante – il s'agit de tout traitement dont le nombre d'opérations n'est pas borné ou dont la complexité est supérieure à $O(n)$. En particulier, toute recherche dans une structure de données dont le nombre d'éléments n'est pas borné doit être faite par dichotomie. Si

le nombre d'éléments dans une structure de données n'est pas borné, alors elle doit être enregistrée sur disque.

Éviter les dépendances externes – le traitement du filtrage doit être effectué uniquement avec des informations locales. Tout appel à l'extérieur génère des dépendances et des comportements non souhaités, en particulier s'il s'agit de ressources dont vous n'avez pas la maîtrise.

Le bon compromis entre efficacité et performance – Généralement les méthodes les plus efficaces sont plus lentes et nécessitent des modèles plus complexes. Nous avons montré dans la Section 3 que les méthodes les plus efficaces dépendent d'une bonne modélisation de la BAL de l'utilisateur. Il est préférable de privilégier, sur la passerelle, les méthodes objectives, dépendant peu des modèles, et implémenter les méthodes plus subjectives sur le serveur hébergeant les BALs.

Fermer les connexions inactives – pour des raisons de performances, assez souvent les spambots[1] ne respectent pas le caractère transactionnel du protocole SMTP et laissent des connexions ouvertes et inactives. Cela occupe des ressources inutiles sur le serveur.

7 « Inside » j-chkmail

j-chkmail est un filtre développé en langage C, utilisant l'API *libmilter* de sendmail. Il intègre plusieurs techniques de filtrage, aussi bien comportementales que de contenu, dans le but d'effectuer un filtrage efficace et rapide. Cet objectif est complété par la protection du serveur de messagerie.

7.1 Limitation de l'utilisation de ressources

Cette catégorie de tests sert soit à protéger le serveur contre l'utilisation abusive par certains clients, comme des tentatives de déni de service. Ceci se fait à deux niveaux : globalement et par adresse IP.

Au niveau global, j-chkmail évalue continuellement le niveau de disponibilité de certaines ressources (charge CPU, nombre de descripteurs de fichiers, ...). Lorsqu'il détecte que le niveau est bas, il accepte alors uniquement des connexions à partir de clients connus. Si le niveau baisse encore, il arrête de recevoir des connexions jusqu'à ce que le niveau de ressources devienne suffisant.

Au niveau individuel, j-chkmail comptabilise des paramètres d'utilisation par adresse IP, soit instantanées, comme par exemple le nombre de connexions ouvertes, ou des paramètres sur une fenêtre glissante. Parmi ces derniers paramètres, on trouve, par exemple, le nombre de connexions, le nombre de messages, le volume transmis, le temps de traitement par la CPU, ...

7.2 Surveillance et commande

La surveillance du filtre est un point très important : il doit être possible de connaître son état en temps réel. Lors du démarrage, j-chkmail ouvre un port (2010, par défaut) et se met en attente de connexions. Il est possible d'utiliser ce port pour se connecter au filtre pour l'interroger sur son état de fonctionnement (cadences diverses, compteurs internes, etc).

Ce canal permet aussi de commander le filtre pour, par exemple, activer ou désactiver des options de filtrage, modifier des valeurs de configuration, etc. Ceci a l'intérêt de ne pas interrompre inutilement le fonctionnement du filtre, comme c'est le cas des autres solutions courantes telles l'envoi de signaux ou le redémarrage du filtre.

Un autre aspect important est la journalisation des événements exceptionnels de filtrage – tout traitement non transparent du filtre doit être enregistré dans un fichier journal de façon à permettre des recherches en cas de réclamation ou d'incident.

7.3 Filtrage viral

La première méthode de filtrage des virus par j-chkmail est la détection de messages avec des fichiers attachés exécutables, identifiés par leur type [13]. C'est la première fonctionnalité implémentée dans j-chkmail. Cette méthode est très efficace, capable de bloquer non seulement la très grosse majorité des virus en circulation actuellement, mais aussi la détection de nouveaux virus, puisqu'elle n'utilise pas les bases de signatures de virus.

Il est aussi possible d'utiliser directement certains scanneurs externes, tels ClamAV, Sophos et autres.

7.4 Filtrage comportemental

Dans la version actuelle de j-chkmail, le filtrage comportemental se limite au comptage d'événements suspects, par client SMTP, sur des fenêtres temporelles glissantes (historique court, moyen ou long), et au rejet des connexions, si la valeur d'un paramètre ou d'une combinaison de paramètres dépasse un seuil.

L'efficacité du filtrage comportemental n'est pas très importante, mais c'est une protection très efficace contre les abus évidents.

En particulier, lors de l'attaque massive des virus MyDoom [14], nous avons bloqué de l'ordre de 8000 virus par jour et, vu les enregistrements de connexions bloquées à cause de leur cadence, nous estimons que le nombre de virus bloqués par ce mode filtrage est du même ordre que ceux bloqués par le filtrage de contenu.

j-chkmail analyse le comportement sur trois niveaux :

- **Historique court** : cet historique, en mémoire contient les informations de synthèse sur toutes les connexions, regroupées par adresse IP des clients SMTP pendant une durée de 20 minutes. Il s'agit de paramètres tels le nombre de connexions, messages, volume, temps de traitement, score des messages, virus, etc. Ce niveau sert surtout à réagir rapidement en cas de besoin et aussi à limiter la consommation de ressources par client, pour ne pas laisser certains clients accaparer les ressources du serveur.
- **Historique moyen** : cet historique, aussi en mémoire, ne contient que les mauvais comportements confirmés, sur une durée de quatre heures : ce sont des paramètres tels le nombre de mauvais destinataires, les messages envoyés à des pièges, messages vides, etc. Les clients dont le nombre de mauvais comportements dépasse un seuil se trouvent bloqués pendant quatre heures.

[1] Spambots ou netbots ou encore zombies sont des ordinateurs, souvent infectés ou piratés, utilisés pour la distribution de spam

- **Historique long** : il s'agit d'un historique sur plusieurs jours résultant soit des observations du filtre ou encore de l'analyse des fichiers de log. Ceci est encore en cours de mise au point.

Cette organisation de l'historique en trois niveaux permet d'éviter le gaspillage de la capacité mémoire. Les informations ne sont enregistrées que pendant la durée dont on a besoin.

7.5 Non-conformité

j-chkmail effectue un nombre important de vérifications de conformité, par rapport, en particulier, aux RFCs 2821 [15] et 2822 [16]. La RFC 2821 concerne surtout les aspects liés à la session SMTP (l'enveloppe et la connexion). tandis que la RFC 2822 concerne le contenu du message (les entêtes et le corps du message).

La conformité de la session concerne des vérifications rapides dont le résultat est soit le rejet de la connexion ou du message ou l'augmentation du score du message. Des exemples de vérification sont la mise en forme et le contenu des paramètres des commandes (EHLO, MAIL, RCPT).

La conformité du message ne génère pas de rejet, mais contribue au score du message. Il s'agit, par exemple, de vérifier si tous les en-têtes obligatoires sont présents, avec le bon nombre et avec la bonne syntaxe.

D'autres vérifications de conformité sont effectuées, comme par exemple, dans les messages ayant une partie du corps en HTML [17], on vérifiera si toutes les balises sont valables.

7.6 Anomalies

Contrairement des non-conformités, les anomalies concernent des points qui ne sont pas forcément interdits par une norme technique. Des exemples de vérification d'anomalie sont :

- Le MX de la partie domaine de l'adresse de l'expéditeur pointe vers une adresse dans un réseau privé ou un serveur inaccessible voire inexistant. Ceci bloque peu de spam (moins de 10 %) mais allège le nombre de bounces[1] dans la file d'attente de messages.
- Le paramètre de la commande EHLO est le nom ou adresse du serveur SMTP et non pas celui du client.
- Les parties MIME *text/plain* et *text/html* du message ne correspondent pas.
- Le message est daté de plus de deux jours.
- ...

7.7 Filtrage de contenu

- **Mots clés et expressions régulières** : Il s'agit de vérifier si des expressions régulières définies dans une liste se trouvent dans le corps du message. Un score est attribué au message, en fonction du poids de chaque expression trouvée dans le message.

  Un des inconvénients de cette méthode est que le message doit être parcouru autant de fois qu'il y a d'expressions définies, ceci à cause de la généralité des expressions.
  Aussi, les caractéristiques des *spams* changeant régulièrement, son utilisation doit être limitée aux expressions dont la durée de vie est longue. La maintenance de cette liste devient prohibitive si l'on cherche à ajouter des nouvelles expressions pour chaque nouveau *spam.*

  En pratique, il est conseillé de ne pas définir plus d'une ou deux centaines d'expressions.

- **Liste noire d'URLs** : Une caractéristique constante des *spams* est la présence d'une URL dans le message, permettant soit d'avoir des informations plus précises sur le « produit », soit d'établir la communication entre le « client » et le spammeur. Le filtrage des URLs consiste à extraire toutes les URLs du message et vérifier leur présence dans une base de données. Contrairement à la recherche de mots clés et expressions régulières, le message n'est parcouru qu'une seule fois et la recherche dans une base de données est très rapide. La base d'URLs utilisée par j-chkmail est celle de surbl.org [11], complétée par des URLs de l'auteur. Cette base contient environ 200000 entrées avec un taux de détection supérieur à 70 % et un taux de faux positifs inférieur à 0,5 %.

  Cette liste est mise à disposition par *surbl.org* sous la forme d'une zone DNS, mais le format préféré par j-chkmail est d'une base de données BerkeleyDB, à cause des temps d'accès : quelques millisecondes, dans le meilleur des cas pour la zone DNS, tandis que si la liste est sur une base de données BerkeleyDB, il est possible d'effectuer plusieurs dizaines de milliers de requêtes par seconde [18]. Sur une machine Sun E280R sous Solaris 9, la vérification d'un message se fait, dans plus de 95 % des cas en moins de 1 ms.

- **Filtrage heuristique (oracle)** : Dans le filtrage heuristique on intègre d'une part les résultats des vérifications de non-conformités et d'anomalies non bloquantes et d'autre part, les critères qui ne sont pas des anomalies proprement dites, mais des indicateurs de *spam*. La présence de certaines balises HTML non souhaitées (IFRAME, IMG, SCRIPT, ...) ou également les messages envoyés par certains MUAs identifiés comme étant des outils d'envoi en masse sont des exemples.

Le résultat du filtrage de contenu s'exprime par un score. Contrairement au filtrage heuristique, la recherche des expressions régulières et le filtrage d'URLs peuvent provoquer le rejet des messages, selon la valeur du score.

S'il n'y a pas de rejet, le score est affiché dans un en-tête du message permettent au destinataire d'effectuer un classement avec ses propres règles et intégrant la valeur du score.

7.8 Greylisting

Le *greylisting* [19] est une méthode de filtrage qui peut être classée aussi bien comme une méthode comportementale et comme une vérification de conformité vis à vis de la RFC 2821. Le principe de la méthode, dans sa version d'origine, est très simple : lorsqu'un client de messagerie reçoit un

[1] Bounce – Message administratif concernant une erreur.

code d'erreur temporaire lorsqu'il essaye de remettre un message, il doit réessayer plus tard. Or, la plupart des *spams* sont distribués par des zombies, qui ne gèrent pas les erreurs, et qui n'essayent pas une deuxième distribution. Lors de la première tentative, le serveur SMTP renvoie une code d'erreur temporaire et enregistre un triplet avec l'adresse IP du client SMTP, l'adresse de l'expéditeur et l'adresse du destinataire. Si le triplet se présente à nouveau un peu plus tard, le message est accepté – les messages suivants avec ce même triplet ne seront plus bloqués.

Mais cette méthode pose deux problèmes majeurs :

- La taille de la base peut devenir importante : nous avons constaté une base dont le nombre usuel d'entrées était entre 500000 et 600000 triplets. Ceci pose des problèmes pour la maintenance périodique et pour la gestion de la base.
- Empoisonnement de la base – il est en effet facile d'attaquer le serveur et d'augmenter, sans limites, le nombre d'entrées : il suffit qu'un client SMTP se connecte 1000 fois et qu'il envoie, à chaque fois, un même message à 500 destinataires différents, pour que 500000 entrées soient créées dans la base de données. Nous avons constaté des attaques de ce genre avec 300000 entrées inutiles, qui ne seraient jamais validées et créées, le même jour, par seulement 9 clients SMTP différents.

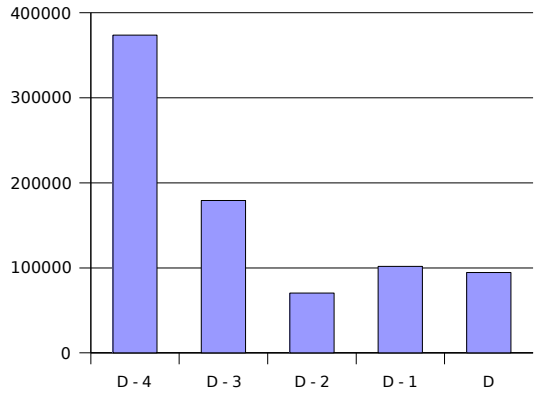

Figure 2 – *Distribution journalière des nouveaux triplets [20]*

En même temps, nous avons constaté que moins de 1 % des triplets âgés de plus de 12 heures étaient validés s'ils restent 3 jours en attente de validation.

j-chkmail apporte quelques solutions originales :

- La durée maximale de vie des entrées n'est pas fixe, comme c'est le cas du greylisting classique, mais dépend d'un facteur de qualité des entrées. Ce facteur de qualité dépend des paramètres de l'entrée elle même, mais aussi des résultats des autres méthodes de filtrage appliqués à ce client SMTP.
- Par une limitation du nombre d'entrées en attente pour chaque client SMTP. Ceci permet de diminuer le risque d'empoisonnement.
- Par l'utilisation de quatre bases de données au lieu de deux, comme c'est le cas pour le *greylisting* classique. Aux deux bases initiales, dont le contenu est essentiellement le même que celui de la version classique, s'ajoute une troisième dont les clés ne sont pas des triplets, mais des couples (adresse client SMTP, domaine de l'expéditeur). Les entrées de cette base sont créées lorsque l'on observe une récurrence d'entrées identiques (même couple), suffisamment âgées et rafraîchies régulièrement. La quatrième base contient essentiellement des adresses IP ayant eu un comportement douteux vis à vis du *greylisting*. La gestion de cette quatrième base n'est pas encore implémentée.

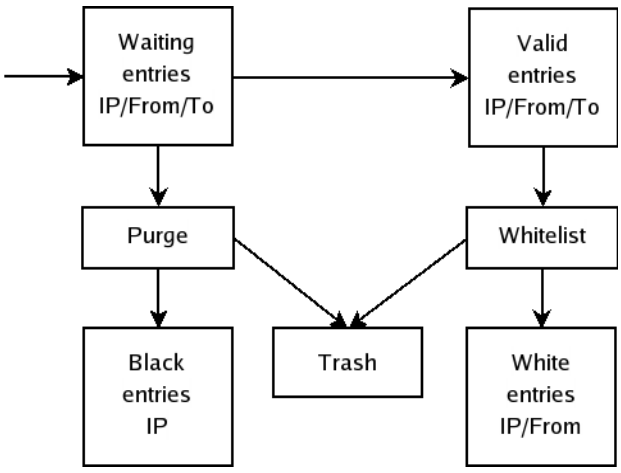

Figure 3 – *Organisation des bases de greylisting dans j-chkmail*

Dans le domaine *ensmp.fr*, le nombre d'entrées usuel dans la base greylisting version classique était de l'ordre de 80000. Avec cette nouvelle organisation, le nombre d'entrées a été réduit à (échantillon typique) :

Entrées en attente	9600
Entrées validées	4604
Liste blanche (IP-domaine)	1597

Table 3 - *Nombre typique d'entrées dans la base greylisting sur le filtre de la passerelle du domaine ensmp.fr*

L'autre particularité du filtrage *greylisting* concerne son utilisation dans une architecture avec plusieurs MXs. Si la synchronisation des bases de données des filtres n'est pas essentielle quand les poids des MXs sont différents, ce besoin est absolu quand les poids sont identiques ou quand il s'agit d'une ferme de machines.

Le filtre *greylisting* de j-chkmail a deux modes de fonctionnement : *standalone* et *client/serveur*.

Dans ce dernier mode, chaque filtre client gère sa propre base de données et, s'il ne trouve pas localement l'information recherchée ou si l'information locale change d'état, il contacte le serveur (*j-greyd*) pour le consulter,

dans le premier cas, ou pour lui notifier, dans le deuxième. Ainsi, chaque client ne conserve que son propre historique sur sa base locale. Le serveur *j-greyd* conserve l'historique de tous les filtres clients.

Les intérêts de cette architecture sont la minimisation des échanges entre les filtres, la disponibilité immédiate de l'information (sans besoin de synchronisations périodiques) et, une fois de plus, l'enregistrement local uniquement des informations utiles. Sur ce dernier point, rappelons qu'une bonne partie des entrées en attente, correspondent à des *spams* : des entrées qui ne seront pas validées.

L'inconvénient est l'existence d'un point faible créé par la localisation de la base centrale sur un serveur unique. Mais cet inconvénient n'est pas aussi grave : les filtres clients continuent à fonctionner avec leurs données locales si le serveur n'est pas accessible et on peut définir plusieurs serveurs.

La version *client/serveur* est encore en cours de validation au moment de la rédaction de cet article.

7.9 Le score des messages et le traitement sur le poste de l'utilisateur

Comme nous avons vu, le traitement du contenu du message et le filtre heuristique attribuent un score aux messages qui ne sont pas bloqués par le serveur. Cet score est matérialisé sur un en-tête sous la forme :

X-j-chkmail-Score: ... 418526F0.000... : XXXX : 5/50 3

Un score non NUL n'est pas un diagnostic binaire de *spam* mais un indicateur. La façon d'utiliser ce score, conseillée par l'auteur, est d'employer deux « *boîtes à spam* » et de mettre dedans les messages avec un score positif, selon sa valeur. Un exemple d'algorithme de classement est :

```
Si expéditeur est connu alors
      met message dans Inbox
sinon si score > 4 alors
      met message dans SCORE-HI
sinon si score > 0 alors
      met message dans SCORE-LO
sinon
      met message dans Inbox
fin si
```

Les messages envoyés par des correspondants connus ou les messages avec un score NUL sont mis dans la BAL d'entrée. Il suffit alors de regarder de temps en temps le contenu des deux « boîtes à spam » avant de les vider. On peut ajuster la valeur du score plus haut de façon à ce que l'apparition de messages légitimes dans la boîte SCORE-HI, soit suffisamment rare.

L'objetif de cette manipulation est de faciliter le classement des messages utiles par le destinataire.

8 L'avenir de j-chkmail

Le développement de j-chkmail se fait par étapes. Une nouvelle version correspond à une ou plusieurs fonctionnalités majeures ajoutées. Les nouveautés majeures de la version en cours sont le *greylisting* et la migration d'une partie de la configuration vers une base de données (j-policy).

Parmi les évolutions en étude pour les prochaines versions, on trouve :

- **Greylisting pour ferme de serveurs ou MXs de même poids** – c'est le mode *client/serveur* décrit dans la section 7.8. Ceci sera intégré dans la version 1.8 définitive ou, au plus tard, dans la version 1.9.
- **Ajout de filtrage bayesien** – quelques expérimentations ont été réalisées avec une base mise au point par Fabrice Prigent à l'Université de Toulouse. Même si les résultats sont moindres que ceux obtenus avec un apprentissage idéal, il est apparu que les résultats qualitatifs obtenus avec une telle transposition sont assez bons pour que son intégration dans j-chkmail soit envisagée comme méthode de filtrage non bloquante.
- **Filtrage heuristique** – Actuellement il y a une trentaine de critères utilisés dans le filtre heuristique. Avec l'intégration du filtre bayesien, les critères moins significatifs doivent disparaître du filtrage heuristique. Quelques uns devront rester, mais comme une possibilité pour l'administrateur du filtre de l'activer ou non.
- **Interface web de configuration** - il s'agit d'un module externe permettant aussi bien de configurer que de commander le fonctionnement de j-chkmail à l'aide d'une interface web.
- **Personnalisation du filtrage par destinataire**
- **Gestion des notifications** – il s'agit de stratégies plus complexes de notification des messages bloqués à cause de virus : notifier une fois par jour au lieu de message par message, notifier seulement les utilisateurs internes, etc.
- **Gestion de la quarantaine** – Il s'agit de permettre à l'utilisateur de libérer, lui même, les messages considérés suspects par le filtre et mis en quarantaine.

Les trois dernières évolutions sont en étude, mais elles sont limitées au filtrage de contenu et aux messages envoyés à un seul destinataire (pas à un alias) ou à plusieurs destinataires mais ayant choisi des options compatibles.

9 Conclusions

Le filtrage de messages sur un serveur traitant un trafic important pour une population hétérogène est un problème difficile, si l'on veut le traiter dans son intégralité. Nous avons présenté les aspects les plus importants qui ont eu une influence dans la conception du filtre j-chkmail.

J-chkmail est un outil de recherche sur le filtrage de messages, mais qui est aussi utilisé en production. Nous estimons à 200 le nombre d'utilisateurs et à 500000 le nombre de boîtes à lettres protégées par ce filtre.

Le plus gros serveur de messagerie connu est pobox.sk. Le débit habituel de chaque passerelle en entrée est de l'ordre de 20K messages à l'heure avec des pics soutenus de 40K messages à l'heure. Des incidents ont été constatés en août 2004 avec 180K connexions dans un intervalle d'une heure. Le serveur, saturé, n'a pas été capable de traiter le trafic arrivant mais il a pu reprendre son activité normale, sans intervention manuelle, aussitôt l'incident terminé.

j-chkmail s'est montré être un outil robuste et fiable même avec des charges soutenues.

La caractéristique explicite « laboratoire de tests » laisse la liberté d'essayer de nouvelles méthodes de filtrage, tout en restant un outil utilisable en production – sans cela l'effort dédié à son développement ne serait pas justifiable.

Bibliographie

[1] B. Page, *A Report on the Internet Worm*, http://www.ee.ryerson.ca:8080/~elf/hack/iworm.html

[2] *Melissa Macro Virus,* http://www.cert.org/advisories/CA-1999-04.html

[3] *Love Letter Worm*, http://www.cert.org/advisories/CA-2000-04.html

[4] S. Aumont et C. Gross – *L'impact de la lutte contre le spam et les virus sur les architectures de messagerie*, JRES 2005, Marseille

[5] *The Bogofilter Home Page*, http://bogofilter.sourceforge.net/

[6] *CRM114 – Frequently Asked Questions,* http://crm114.sourceforge.net/FAQ.txt

[7] The DSPAM Home Page, http://dspam.nuclearelephant.com/

[8] *The Apache SpamAssassin Project*, http://spamassassin.apache.org

[9] Willis W. Harman, *Principles of the Statistical Theory of Communications,* 1963, McGraw-Hill Book Co.

[10] P.G. Hoel, S.C. Port et C.J.Stone, *Introduction to Probability Theory*, 1971, Houghton Mifflin Company

[11] *SURBL – Spam URI Realtime Blocklists*, http://www.surbl.org

[12] *MAPS – mail-abuse.com*, http://www.mail-abuse.com

[13] J.M M.Cruz et G. Huberman*, Le filtrage de mail sur un serveur de messagerie avec j-chkmail*, JRES 2003, Lille, http://2003.jres.org/actes/paper.32.pdf

[14] *Technical Cyber Security Alert TA04-028A - W32/MyDoom.B Virus*, http://www.us-cert.gov/cas/techalerts/TA04-028A.html

[15] *RFC 2821 – Simple Mail Transfer Protocol*, 2001, http://www.ietf.org/rfc/rfc2821.txt

[16] *RFC 2822 – Internet Message Format*, 2001, http://www.ietf.org/rfc/rfc2822.txt

[17] *HyperText Markup Language (HTML) Home Page*, http://www.w3.org/MarkUp/

[18] C. Assmann, *sendmail X : Requirements, Architecture, Functional Specification, Implementation and Performance*, May 2005, Emeryville CA

[19] E. Harris, *The Next Step in the Spam Control War*, http://projects.puremagic.com/greylisting/

[20] J.M. M.Cruz, *Mail filtering on medium/huge mail servers with j-chkmail*, Terena Networking Conference 2005, Poznan, http://www.terena.nl/conferences/tnc2005/