La messagerie Internet

Infrastructure
Protocoles
Administration
Evolutions

pascal.malterre@cea.fr

Historique (1/3)

- La préhistoire...
  - Envoi de messages entre différents utilisateurs des mainframes à la fin des années 60 (MAILBOX)
  - Premier envoi d'un message entre deux machines du réseau ARPANET courant 1972 (1ère utilisation du caractère @)
  - En 1973, les mails représentent ¾ du trafic ARPANET
  - Nombreux problèmes d'interropérabilité (RFC 680, 724, 733, etc.)
  - Eric Allman développe *delivermail* (utilisation de *FTP over NCP* pour transmettre les messages)

Historique (2/3)

- Les temps anciens (... ou l'avènement de TCP/IP)
  - Avant la naissance du protocole SMTP, la transmission des mails se fait au moyen de FTP, UUCP, etc.
  - Eric Allman développe *sendmail*
  - La RFC 821 donne les premières spécifications du protocole SMTP (Simple Mail Transfer Protocol) (en 1982)
  - La RFC 822 spécifie le format des messages

Historique (3/3)

- Les temps modernes...
  - L'évolution vers la messagerie actuelle :
    - Encapsulation de contenus multimédias (MIME)
    - Interconnexion avec les annuaires LDAP, Active Directory
  - Les besoins émergents en terme de sécurité :
    - Authentification, confidentialité, etc.
    - Filtrage applicatif : spam, antivirus, etc.

Vocabulaire

- MTA (Mail Transfer Agent)
  - Programme responsable de l'acheminement des messages
- MUA (Mail User Agent)
  - Programme permettant à l'utilisateur de lire et d'envoyer des messages
    - Dialogue avec le MTA via le protocole SMTP (client uniquement) et avec le serveur de boîtes aux lettres via POP ou IMAP
- MDA (Mail Delivery Agent)
  - Interface entre le MTA et la boîte aux lettres de l'utilisateur

MUA et MDA

- Les MUA sont les programmes utilisés par les utilisateurs pour gérer leur messagerie au quotidien
  - Exemples : Microsoft Outlook, Evolution, Thunderbird, Mutt, Pine, mailx, etc.
- Un MDA est généralement directement exécuté par le MTA afin d'ajouter le message à la BAL de l'utilisateur destinataire. C'est en fait le MDA qui prend alors en charge la partie finale du traitement
  - Exemples : procmail, maildrop, etc.

Le format des adresses

- Une adresse mail est une chaîne de caractère contenant le symbole "@"
  - De la forme : "**local-part@domain**" (la partie "domain" est en fait tout ce qu'il y a après le dernier "@", elle est indépendante de la casse des caractères)
  - Caractères ASCII uniquement
- Adresses particulières
  - postmaster@domain.com (RFC 822)
  - abuse@domain.com
  - ...

Le format des messages

- Un message est constitué de trois parties :
  - L'enveloppe est utilisée par le MTA pour l'acheminement du courrier, il s'agit essentiellement de l'expéditeur et du destinataire
  - Les en-têtes (ou *headers*) sont utilisés par les MUA
  - Le corps est le contenu du message envoyé au destinataire

Le format des messages (2/3)

- Le format et l'interprétation des en-têtes sont spécifiés dans la RFC 822 (1982)
  - « *822 allows much more flexibility than a typical mail-reading program can actually handle; meanwhile it imposes restrictions that millions of users violate every day* »
- Les en-têtes sont de la forme « **Nom: valeur** »

```
Date: Sat, 21 Aug 2004 12:27:41 +0200
X-Spam-Flag: YES
From: Bob Marley <bob@marley.com>
To: Pascal Malterre <pascal@garance.fr>
Subject: Fw: failure notice

Humm... ?!
```

Le routage des mails et le DNS

- Problème : à quel serveur doit-on s'adresser pour envoyer un mail destiné à "**local@domaine.com**"
  - Le MTA effectue une requête DNS afin de connaître l'enregistrement MX de "domaine.com"

```
...
bash$ host -t mx rstack.org
rstack.org is handled by 20 mrelay1.online.net
rstack.org is handled by 20 mrelay2.online.net
rstack.org is handled by 20 mrelay3.online.net
rstack.org is handled by 20 mrelay4.online.net
rstack.org is handled by 10 mx.online.net
```

  - En cas d'échec de la requête DNS, cela ne veut pas dire qu'il n'y a pas d'enregistrement MX pour le domaine concerné : le client doit réessayer un peu plus tard...
  - Si la requête DNS ne renvoie aucun enregistrement MX, alors le MTA doit prendre comme MX l'hôte lui-même

Le routage des mails et le DNS (2/2)

- Le serveur prend ensuite la liste des serveurs destinataires par ordre croissant de préférence et se connecte ensuite sur le port TCP/25 pour envoyer le message par SMTP
  - En cas d'échec **permanent**, le client retourne un message d'erreur à l'expéditeur (*bounce message*)
  - En cas d'échec temporaire, le client réessaye d'envoyer le message en utilisant les serveurs de mails ayant une priorité inférieure. S'il ne restent plus de serveurs dans la liste, le message est alors mis en attente
- L'utilisation des MX est spécifiée dans la RFC 974

Le protocole SMTP

- SMTP = Simple Mail Transfer Protocol
  - Définit la manière de communiquer entre deux MTA en utilisant une connexion TCP
    - Protocole de type « PUSH »
  - Utilise un alphabet ASCII 7 bits (en TCP, transmission de 8 bits avec le bit de poid fort à 0)
  - Le nombre de commandes utilisées est relativement faible (inférieur à 12)
  - Le protocole est défini dans la RFC 821 (1982)

Les principales commandes SMTP (1/5)

- La commande HELO permet à la machine source de s'identifier auprès du serveur SMTP

```
HELO mail.rstack.org
```

```
ie. "Salut, je suis mail.rstack.org"
```

  - D'après la RFC 1123, le paramètre transmis doit être un nom d'hôte valide au sens DNS
  - La responsabilité est à la charge de l'émetteur
  - Bien que cela soit interdit par la RFC 1123, certaines implémentations de serveurs SMTP refusent les mails d'après certaines vérifications sur la commande HELO (reverse DNS, MX, etc.).

Les principales commandes SMTP (2/5)

- La commande MAIL permet de spécifier l'adresse de l'expéditeur (*return-path*) et de réinitialiser la liste des destinataires
  - La commande MAIL est suivie de la chaîne "FROM:" et d'une adresse de retour (la chaîne <> représente une adresse vide) et éventuellement des informations supplémentaires (8BITMIME)

```
...
MAIL FROM:<pascal@rstack.org>
250 ok
```

Les principales commandes SMTP (3/5)

- La commande RCPT permet d'ajouter une adresse à la liste courante des destinataires
  - Cette commande est suivie de la chaîne "TO:", d'une adresse et éventuellement d'informations optionnelles
  - Plusieurs commandes RCPT peuvent être utilisées à la suite les unes des autres

```
...
RCPT TO:<webmaster@rstack.org>
250 ok
RCPT TO:<moutane@rstack.org>
250 ok
```

Les principales commandes SMTP (4/5)

- La commande DATA permet de transmettre le message
  - Cette commande n'accepte pas de paramètre
  - La transmission du message commence immédiatement après la réponse du serveur
  - Chaque ligne du message se termine par <CRLF>, la fin est signalée par la séquence <CRLF>.<CRLF>

```
...
DATA
354 go ahead

Hello world !

.
250 ok 1102890861 qp 5433
```

Les principales commandes SMTP (5/5)

- La commande VRFY permet de vérifier la validité d'une adresse
  - Problème de confidentialité vis à vis du Spam
  - La RFC 1123 définit un nouveau code d'erreur (252) permettant de traiter la commande VRFY
- La commande EXPN permet de développer une adresse générique représentant une liste de diffusion
  - Le terme "liste de diffusion" peut également désigner un utilisateur local (par exemple : "expn root")
  - Généralement non implémentée (cf. Spam)

Les évolutions du protocole SMTP

- Extended SMTP (ESMTP)
  - Ces extensions (définies dans la RFC 1425) maintiennent néanmoins une compatibilité ascendante
  - Utilisation de EHLO à la place de HELO, puis code réponse 250

```
bash$ nc smtp.wanadoo.fr 25
220 mwinf0209.wanadoo.fr ESMTP **************************
EHLO marley.com
250-mwinf0209.wanadoo.fr
250-PIPELINING
250-SIZE 10485760
250 8BITMIME
```

Remarque sur les serveurs SMTP

- Lorsqu'un serveur de mail accepte un message, il ajoute un en-tête "**`Received:`**" au début du message
- Le protocole TCP n'est utilisé qu'en half-duplex
  - Le client envoie une requête, attend la réponse, etc.
  - Importance de la rapidité de réponse pour les serveurs
- Généralement, les serveurs SMTP ne parsent pas les headers d'un message, excepté pour compter le nombre de saut (nb. de "**`Received:`**")

Remarque

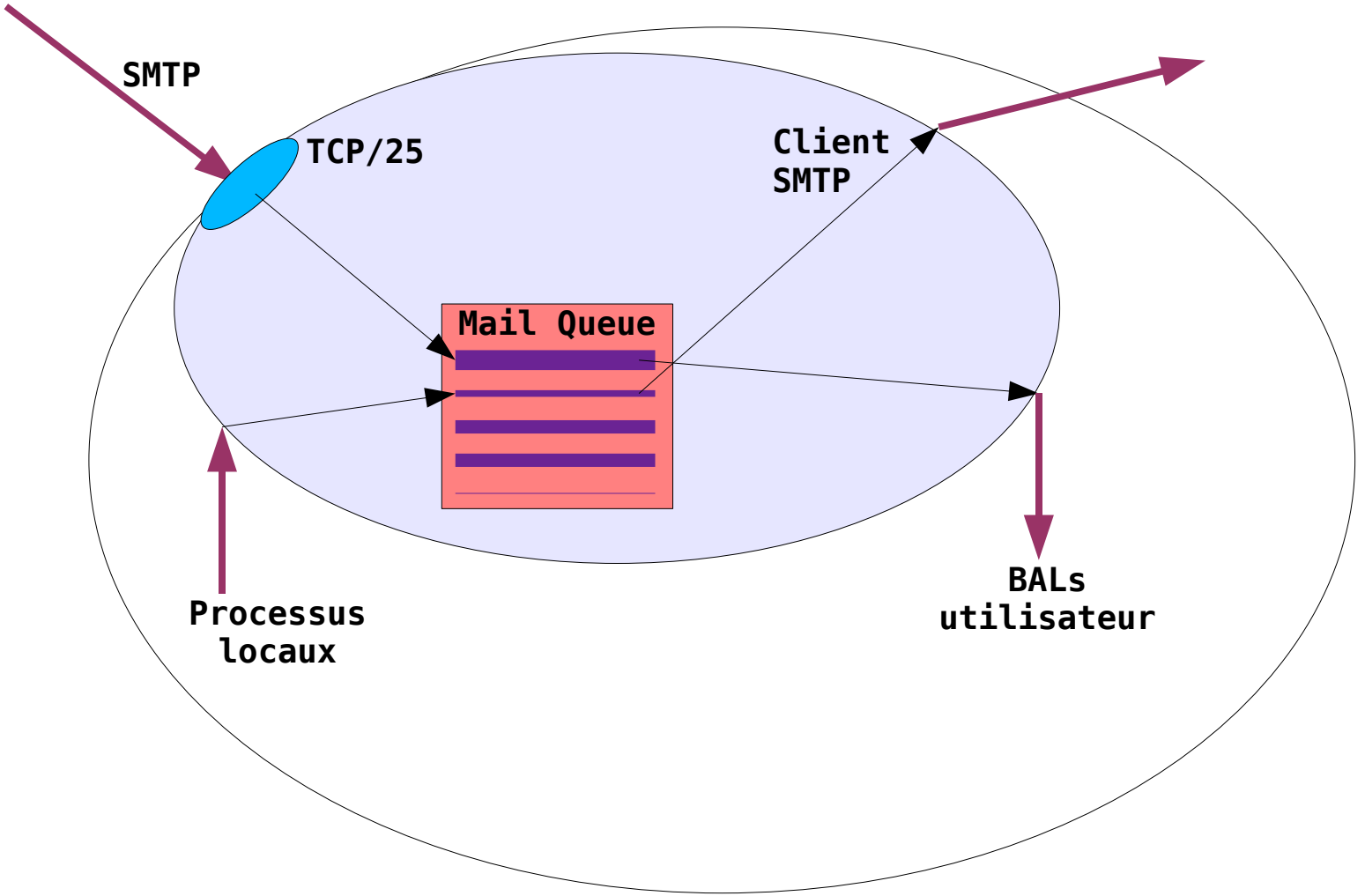
- Dans le paragraphe 5.3.3 de la RFC 1123, intitulé "*Reliable Mail Receipt*", il est écrit :

  « *When the receiver-SMTP accepts a piece of mail (by sending a "250 OK" message in response to DATA), it is accepting responsibility for delivering or relaying the message. It must take this responsibility seriously, i.e., it MUST NOT lose the message for frivolous reasons, e.g., because the host later crashes or because of a predictable resource shortage* »

Les objectifs d'un serveur de mail

- Accepter les mails issus des processus locaux (programmes et/ou utilisateurs)
- Accepter les mails provenant d'autres serveurs
- Transmettre les mails aux processus locaux
- Transmettre les mails aux autres serveurs

L'architecture d'un serveur de mail



Quelques exemples de MTA

- Sendmail ("le" mailer Unix historique)
  - Approche monolitique, multiples failles de sécurité, configuration complexe, etc.
- qmail
  - Architecture modulaire conçue de façon très sécurisée
  - Configuration aisée, très performant
- Postfix
  - Architecture similaire à qmail
  - Plus moderne (plus d'évolutions possibles)
- Exim
  - ?

Accès aux boîtes aux lettres : POP3 (1/2)

- POP3 = Post Office Protocol version 3
  - RFC 1939
  - Protocole permettant à un programme local de récupérer les mails stockés dans une BAL *distante* (ie. présente sur un serveur distant)
- Utilise le port TCP/110
- Encore majoritairement utilisé par les ISP :
    - Le protocole est très simple, donc peu d'incompatibilités avec les programmes clients
    - Le fonctionnement par défaut favorise le contrôle de la taille des boites aux lettres
    - Convient parfaitement à un accès à Internet non-permanent

Accès aux boîtes aux lettres : POP3 (2/2)

- Les mécanismes d'authentification POP3
  - Comme beaucoup de protocoles un peu anciens, le mot de passe est transmis en clair par défaut
- La commande (optionnelle) APOP
  - Permet une authentification par challenge/response
  - Le serveur envoie une chaîne contenant (entre-autre) un marqueur de temps dans sa bannière de connexion
    - De la forme : <marqueur-unique@nom-serveur>
  - Le client concatène ce marqueur et son mot de passe et renvoie le résultat au serveur qui peut ainsi vérifier son identité

Accès aux boîtes aux lettres : POP3 (3/3)

- Quelques exemples...

```
$ nc pop.free.fr 110
+OK <14876.1102792612@pop2-q.free.fr>
+OK <14876.1102792612@pop6-g25.free.fr>

$ nc pop.wanadoo.fr 110
+OK connected to pop3 on 0401
+OK connected to pop3 on 0404

$ nc pop.laposte.net 110
+OK POP3 server ready (7.0.028) <7FA75DABA11.....068EF32B2625@mx.laposte.net>
+OK POP3 server ready (7.2.060.1) <73D3317A9A..............@mx.laposte.net>

$ nc pop.tiscali.fr 110
+OK POP3 server ready (7.1.026) <04B411DC552.....C9F4118@mail.libertysurf.net>
+OK POP3 server ready (7.1.026) <8500B05B9F7............@mail.libertysurf.net>

$ nc pop.club-internet.fr 110
+OK POP3 FLASH Server
+OK POP3 FLASH Server
```

Accès aux boîtes aux lettres : POP3 (4/4)

- Quelques extensions au protocole POP
  - POP over SSL (commande STARTTLS)
  - Autres mécanismes d'authentification (SASL)
  - *Login delay*

Accès aux boîtes aux lettres : IMAP (1/3)

- IMAP = Internet Message Access Protocol
  - La version courante est la version 4 révision 1 généralement nommée IMAP4rev1 (RFC 3501)
- Caractéristiques
  - TCP/143 (ou TCP/993 pour *imap over SSL*)
  - Les messages restent sur le serveur, l'utilisation se fait généralement en mode connecté
  - Boîtes aux lettres partagées, accès concurrentiels, etc.
  - Création de dossiers IMAP résidant sur le serveur

Accès aux boîtes aux lettres : IMAP (2/3)

- Contraintes d'exploitation
  - Gourmand en ressources
    - Au niveau système, il s'agit d'un protocole complexe et certaines fonctionnalités (s'exécutant sur le serveur) peuvent être coûteuses (tri, threading, etc.)
    - Maîtrise de la taille des BALs
    - Au niveau réseau, de multiples connexions IMAP peuvent être ouvertes simultanément lors de l'accès à une BAL
  - La compatibilité avec le protocole peut se révéler très différente entre les divers clients de messagerie

Accès aux boîtes aux lettres : IMAP (3/3)

- Quelques exemples de serveurs IMAP :
  - UW Imap
  - Courier IMAP
  - Cyrus IMAP Server
  - Microsoft Exchange Server
  - Dovecot
  - ...

Les principaux formats de BAL

- Deux principales approches :
  - Le format de la boîte aux lettres est complètement masqué pour l'utilisateur, et ce dernier n'accède à la boîte qu'au travers d'un protocole ou un client spécifique
    - Microsoft Exchange, Cyrus IMAPd
  - Le format de la BAL est complètement standard et cette dernière peut être utilisée par différents MDA et/ou MUA
    - Meilleure interropérabilité
- Deux principaux types couramment utilisés :
  - Maildir
  - mbox

Les principaux formats de BAL : mbox

- Format traditionnel sous Unix
  - Plusieurs variantes : mboxrd, mboxo, mboxcl, etc.
- Un seul fichier est utilisé pour stocker tous les messages présents dans la BAL
- Chaque message débute par la chaine « **`From`** » suivi du message (en-têtes et corps), puis d'une ligne vide
- Utilisation de verrous pour les accès en écriture à la boîte aux lettres, nombreux problèmes :
  - Accès à la boîte par plusieurs processus, intégrité des messages en cas de crash, utilisation de NFS, etc.

Les principaux formats de BAL : Maildir

- Format popularisé par qmail
- Un mail est enregistré dans un fichier, sans aucune modification (conforme à la RFC 822)
- Une BAL au format Maildir contient 3 sous-répertoires :
  - **`new/`** contient les nouveaux messages, ils ne sont pas encore lus par l'utilisateur
  - **`cur/`** contient les messages actuels (déjà lus)
  - **`tmp/`** contient les messages en cours d'élaboration
- Avantage : accès simultanés à la BAL, NFS, etc.
- Problème : générer des noms de fichiers uniques

Le problème du Spam (1/6)

- pourriel, pollupostage, etc.
  - « *courrier électronique non sollicité, envoyé le plus souvent massivement, sans l'accord des destinataires* »
  - Représente 75% du trafic mail sur internet (plusieurs centaines de millions de mails/jour)
- Le Spam est avant tout un problème économique
  - Le coût d'envoi d'un message est très faible et le stockage des messages est à la charge des intermédiaires et du destinataire
- A l'heure actuelle, il n'existe aucune solution définitive au problème du Spam
  - Une multitude de techniques peuvent néanmoins être utilisées
  - Le nombre de spams a cessé de croître !

Le problème du Spam (2/6)

- Différents types de nuisances peuvent être considérées comme du Spam :
  - Informations à caractère commercial
    - Entreprises peu scrupuleuses vis-à-vis de la Netiquette
  - Escroqueries diverses
    - Pornographie, médicaments, crédit financier, etc.
  - Phishing (*password harvesting fishing*)
    - Obtention d'informations confidentielles (ex. : numéro de carte bancaire) en se faisant passer auprès des victimes pour quelqu'un digne de confiance (autre nom : scam)
  - Hoax (ou canular informatique)

Le problème du Spam (3/6)

- Vérifications DNS
  - Requête de type MX sur le domaine de l'adresse mail de l'expéditeur
  - Recherche du domaine à partir de l'adresse IP et comparaison avec le champ *From:* du message
- Dans les 2 cas, les risques de faux-positifs sont très importants
- Interropérabilité

Le problème du Spam (4/6)

- Black list IP
- RBLs (Realtime Blackhole Lists)
  - Principe : pour chaque mail entrant, rechercher l'adresse IP de l'expéditeur dans des listes publiques d'adresses IP de *spammers* connus
  - S'appuient généralement sur le DNS (DNSRBLs)
  - Le déploiement est simple, relativement peu coûteux en terme de ressources CPU, et l'impact au niveau réseau reste faible
  - Inconvénients :
    - Risques importants de faux-positifs
    - La mise à jour des listes publiques est une intervention humaine

Le problème du Spam (5/6)

- Le *Greylisting*
  - Technique relativement récente et assez efficace
  - Codes d'erreurs SMTP (RFC 821) :
    - { 1yz, 2yz, 3yz } : Positive Reply
    - 4yz : Transient Negative Completion Reply
    - 5yz : Permanent Negative Completion Reply
  - Les sources de Spam ne prennent pas en compte les erreurs SMTP temporaires
  - Fontionne aussi pour les virus de messagerie
- Autres : smtp-delay, etc.

Le problème du Spam (6/6)

- Mise en place d'un filtrage applicatif
  - Recherche de mot-clés spécifiques
    - Efficace mais facilement contournable, par exemple remplacer le terme *Viagra* par *V1agra*
  - « Rule-based scoring systems »
    - Systèmes capables d'éliminer 90% du Spam
    - Exemple : SpamAssassin
  - Filtrage basé sur des méthodes probabilistes
    - Filtrage bayésien

Sender Policy Framework

- Le responsable d'un domaine de messagerie publie dans le DNS (RR TXT) la liste des machines de son domaine autorisées à envoyer des mails
- Lors de la réception d'un mail, le MTA vérifie le domaine présent dans MAIL FROM
  - Par exemple, un message ayant un MAIL FROM positionné à *@domaine.com entraînera une vérification sur les relais autorisés pour domaine.com
  - Interessant pour lutter contre les MAIL FROM forgés
- Problèmes : forwards, MAIL FROM vide, accès nomades, interropérabilité, etc.

Domain Keys Identified Mail

- Système de signature sur les entêtes et le corps d'un message par ajout d'un header spécifique (DKIM-Signature)
  - Clé publique récupéré après une requête DNS
  - Transparent par rapport au routage des mails en SMTP
- Lutte efficace contre le phising
- Inconvénients :
  - L'enveloppe n'est pas prise en compte dans la signature
    - Pas de protection contre le rejeu
  - Modification en cours de route du message

Quelques caractéristiques de SpamAssassin (1/3)

- SpamAssassin
  - Développé en PERL, la version courante est 3.1.0
  - Peut s'intégrer à la fois dans un environnement serveur (ie. couplé à un MTA) ou utilisateur (procmail)
- Idée directive
  - Appliquer une multitude de techniques différentes puis corréler, pondérer, et additionner l'ensemble des résultats
  - Si le résultat final dépasse un certain seuil, le message est considéré comme un Spam
  - Le message est simplement marqué comme "Spam" au moyen de différents headers : "**X-Spam-FLAG**", etc.

Quelques caractéristiques de SpamAssassin (2/3)

- Caractéristiques techniques :
  - Utilise différents tests DNS, RBLDNS, etc. La pondération des résultats diminue considérablement le nombre de faux-positifs
  - Intègre un système de type "Rule-based Scoring" prenant en compte l'intégralité du message, c'est à dire le contenu et les éléments structurels
  - Gestion de white-lists, black-lists, ainsi qu'un système d'Auto White-Lists (AWL)
  - Filtrage Bayésien

Quelques caractéristiques de SpamAssassin (3/3)

- Intégration dans un environnement serveur
  - Coûteux en terme de ressources systèmes (PERL)
  - Fonctionnement en mode client/serveur
    - Afin d'éviter le chargement des règles à chaque analyse d'un nouveau message, le daemon spamd est utilisé. Ce dernier est interrogé par un client léger (spamc) au travers d'une socket Unix ou TCP
  - Intégration possible à deux niveaux :
    - Au niveau du process serveur SMTP (mise en place d'un filtre global)
    - Au niveau du process de *mail delivery* (gestion de la configuration du filtre par utilisateur)

Filtrage antivirus

- Le fonctionnement est souvent similaire à celui vu précédemment
- Fonctionnement en mode Client-Serveur
  - Une machine dédiée peut se révéler nécessaire (un filtrage antivirus est gourmand en ressources CPU)
- Contraintes d'exploitation
  - Gestion de la quarantaine sur des serveurs internet (?)
  - Mise à jour de la base de signatures
- Exemple concret : ClamAV