Université Panthéon-Assas

école doctorale d'informatique (455)

Thèse de doctorat en
soutenue le 9 décembre 2011

Thèse de Doctorat / Décembre 2011

RETABLIR LA CONFIANCE DANS LES MESSAGES ELECTRONIQUES

Le traitement des causes du "spam"

Université Panthéon-Assas

Auteur

Eric LAURENT-RICARD

Sous la direction de Monsieur le Professeur David NACCACHE

Membres du jury :

Messieurs les Professeurs Jean DONIO, David NACCACHE, Jean-Jacques QUISQUATER et Harald WERTZ

Avertissement

La Faculté n'entend donner aucune approbation ni improbation aux opinions émises dans cette thèse ; ces opinions doivent être considérées comme propres à leur auteur.

Remerciements

Je tiens tout d'abord à remercier la vie,

La vie qui m'a permis de réaliser nombre de mes objectifs,

La vie qui m'a emporté dans un tourbillon et m'a porté jusqu'ici,

La vie qui fait naître l'Esprit puis le libère,

La vie qui est un chemin, et non un but, et qui permet d'avancer.

Sur ce chemin j'ai rencontré de nombreuses personnes, et plus particulièrement celles qui m'ont formé, aidé ou apporté leur soutien et que je souhaite remercier ici.

Ne pouvant les nommer toutes, je me restreindrai à quelques proches :

Ma famille qui m'a toujours soutenu et encouragé, me donnant la force d'entreprendre pour l'avenir. La « transmission » n'est pas un vain mot…

Mon mentor, Monsieur Jean DONIO, sans qui je n'aurais pu arriver où je suis,

Et bien sur, mon Directeur de Thèse, Monsieur David NACCACHE, mon confrère, mon tuteur et surtout mon ami…

Que tous les autres ne pensent pas que je les oublie, mais l'espace et le temps me manquent pour les évoquer.

Eric LAURENT-RICARD

Résumé :

L'utilisation grandissante de la messagerie électronique dans les échanges dématérialisés, aussi bien pour les entreprises que pour les personnes physiques, et l'augmentation du nombre de courriers indésirables, nommés « *spams* » (pourriels) génèrent une perte de temps importante de traitement manuel, et un manque de confiance à la fois dans les informations transmises et dans les émetteurs de ces messages.

- Quels sont les solutions pour rétablir ou établir la confiance dans ces échanges ?
- Comment traiter et faire diminuer le nombre grandissant de « *spams* » ?

Les solutions existantes sont parfois lourdes à mettre en œuvre ou relativement peu efficaces et s'occupent essentiellement de traiter les effets du « *spam* », en oubliant d'analyser et de traiter les causes.

L'identification, si ce n'est l'authentification de l'émetteur et des destinataires, est un des points clés permettant de valider l'origine d'un message et d'en garantir le contenu, aussi bien qu'un niveau important de traçabilité, mais ce n'est pas le seul, et les mécanismes de base mêmes de la messagerie électronique, plus précisément au niveau des protocoles de communication sont également en jeu.

Le contenu de cette thèse portera plus spécifiquement sur les possibilités liées aux modifications de certains protocoles de l'Internet, en particulier le protocole SMTP, la mise en œuvre de spécifications peu utilisées, et les outils et méthodes envisageables pour garantir l'identification des parties de façon simple et transparente pour les utilisateurs.

L'objectif est de définir, d'une part une méthodologie d'utilisation de la messagerie pouvant assurer fiabilité et confiance, et d'autre part de rédiger les bases logiques de programmes clients et serveurs pour la mise en application de cette méthodologie.

Descripteurs :

«spam» ; pourriel ; e-mail ; confiance ; identification ; authentification ; SMTP ;

Title and Abstract :

RESTORING CONFIDENCE IN ELECTRONIC MAILS

The growing use of email in dematerialized exchanges, for both businesses and individuals, and the increase of undesirable mails, called "spam" (junk emails) generate a significant loss of time of manual processing And a lack of confidence both in the information transmitted and the issuers of such messages.

- What are the solutions to restore or build confidence in these exchanges?
- How to treat and reduce the growing number of «spam»?

Existing solutions are often cumbersome to implement or relatively ineffective and are primarily concerned with treating the effects of "«spam»", forgetting to analyze and address the causes.

The identification, if not the authentication, of the sender and recipients, is a key point to validate the origin of a message and ensure the content, as well as a significant level of traceability, but it is not the only one, and the basic mechanisms, themselves, of the email system, more precisely in terms of communication protocols are also at stake.

The content of this thesis will focus primarily on opportunities related to changes in some Internet protocols, in particular SMTP, implementation specifications rarely used, and the tools and possible methods to ensure the identification of parties in a simple and transparent way for users.

The objective is to define, firstly a methodology for using the mail with reliability and confidence, and secondly to draw the logical foundations of client and server programs for the implementation of this methodology.

Keywords :

«spam» ; Junk emails ; confidence ; authentication ; identification ; SMTP

Principales abréviations

SMTP : Simple Mail Transfer Protocol
POP : Post Office Protocol
IMAP : Internet Message Access Protocol
RFC : Request For Comment
MUA : Message User Agent
MTA : Message Transfer Agent
MDA : Message Delivery Agent
MSA : Message Submission Agent
TCP/IP : Transport Control Protocol over Internet Protocol
LDAP : Lightweight Directory Access Protocol
DNS : Domain Name System
SHA1 : Secured Hash Algorithm V1
MD5 : Message Digest V5
NTP : Network Time Protocol
AC : Autorité de Certification
AO : Opérateur de Certification
AE : Autorité d'Enregistrement
PKI : Public Key Infrastructure
ICP : Infrastructure à Clés Publiques
DKIM : Domain Keys Identified Mail
TLS / SSL : Transport Layer Security / Secure Socket Layer

Sommaire

I. Introduction

La messagerie électronique est un outil qui est devenu indispensable, à la fois dans les échanges professionnels et les échanges privés.

J'en étais déjà convaincu vers la fin des années 80, avant même que je ne commence à développer la première société qui a connecté des entreprises en France à l'Internet en 1993 (EUnet France).

A cette époque, l'utilisation de la messagerie électronique était encore balbutiante, et même dans notre environnement spécialisé, nous ne recevions pas plus d'une dizaine de messages par jour, tous ayant naturellement leur importance.

La rapide croissance de l'utilisation de l'Internet a pris de court les passionnés qui amélioraient sans cesse les protocoles et outils permettant d'échanger sur ce réseau.

En conséquence, de nombreuses organisations ont su exploiter les faiblesses de ce système pour « inonder » les utilisateurs de la messagerie électronique de messages indésirables, à tel enseigne que 17 ans plus tard, il devient parfois difficile de retrouver les messages « utiles » parmi ceux que nous recevons !

Certes, une activité florissante a été créée pour répondre, au moins partiellement, à ces besoins, et de nombreuses sociétés vantant les mérites de leurs solutions *« anti-spam» »* avec un succès modéré au niveau des résultats, ont vu le jour au cours de ces années.

Néanmoins, ces solutions ne traitent que les effets du *« spam »* en triant les messages qui arrivent, mais pratiquement aucune solution n'empêche les émetteurs de déposer dans nos boites aux lettres leurs messages publicitaires.

Ayant travaillé de nombreuses années avec ces technologies, j'ai pu analyser et comprendre quels mécanismes pourraient être mis en place pour tenter de répondre à ces contraintes.

C'est un sujet auquel je réfléchis depuis plusieurs années pour affiner, tant la compréhension que l'analyse des méthodes utilisées par les *« spammeurs »*, en cherchant des moyens utiles pour améliorer cet environnement de la messagerie électronique qui est devenu un centre de communication névralgique au sein de l'entreprise, et un outil pratiquement indispensable pour chacun dans sa vie privée.

La directive Européenne du 13 Décembre 1999, sur la signature électronique, et sa transposition en droit Français par la Loi du 13 Mars 2000, ont permis de créer un cadre juridique initial sur lequel il est intéressant de s'appuyer.

Ainsi, en partant d'une analyse des **faiblesses actuelles de la messagerie électronique,** et en intégrant une analyse technique de ses mécanismes avec des notions juridiques de l'écrit et de la signature électronique, il est possible de concevoir une nouvelle approche de la problématique liée à la confiance dans les échanges de messages électroniques, sans tomber dans la « lourdeur » et le coût de solutions professionnelles sécurisées utilisées dans certains environnements d'entreprises.

Une nouvelle architecture de confiance peut se mettre en place en apportant une valeur aux messages, une éventuelle confidentialité et une réduction notable du « spam ».

Contrairement au dicton, il est parfois utile de « réinventer la roue »…

II. Les causes du manque de confiance dans la messagerie

Le besoin de confiance dans les échanges électroniques devient essentiel compte tenu du développement de ceux-ci.

Néanmoins, la croissance du « spam » et de nombreuses faiblesses des protocoles de messagerie et de leurs implémentations font partie des causes principales de la défiance que rencontrée actuellement vis-à-vis de la messagerie électronique.

Cette défiance arrive à un point tel que plusieurs sociétés ont décidé de ne plus utiliser la messagerie électronique dans leurs échanges internes, et d'y préférer des mécanismes d'échanges en temps réel comme le « chat » (clavardage).

Rejeter un système pour la seule raison qu'il possède des faiblesses n'est pas une solution viable à long terme, et **il convient de se pencher sur les causes de cette défiance afin d'envisager des solutions pour les éliminer**.

1. LE « SPAM »

Ce terme a été utilisé pour qualifier les messages électroniques indésirables reçus dans nos boîtes aux lettres électroniques. Une de ses traductions Française est « pourriel ».

Le principe est exactement le même que la distribution de tracts publicitaires papiers dans nos boîtes aux lettres du monde matériel.

Imaginez votre boîte aux lettres débordant de papiers publicitaires à tel enseigne que la recherche du seul courrier important dans cet ensemble prenne des heures !

Dans le monde dématérialisé de l'Internet ces envois non sollicités prennent des proportions gigantesques, et rapidement les boîtes de messagerie sont saturées de messages indésirables, car l'envoi d'un e-mail est peu coûteux et facile à réaliser en nombre.

2. ABSENCE D'AUTHENTIFICATION

Actuellement, aucune solution, au niveau des protocoles de l'Internet, ne propose une authentification de l'émetteur des messages électroniques.

Seules des fonctions et applications complémentaires, parfois mal intégrées, gèrent cette authentification à l'aide d'un certificat de signature électronique.

Des solutions anti-phishing[1] comme DKIM[2] permettent l'authentification du nom du domaine de l'émetteur et l'association entre celui-ci et le serveur de messagerie associé.

C'est un début de solution, mais qui est limité et très insuffisant pour instaurer la confiance.

3. ENVOI DES MOTS DE PASSE EN CLAIR

Les extensions de sécurité du protocole SMTP permettent d'établir un lien sécurisé (TLS/SSL) mais seulement de façon optionnelle, et la plupart des ISP n'obligent pas à sa mise en place.

[1] Phishing : hameçonnage
[2] Domain Key Identified Mail

En conséquence, les mots de passe d'accès aux serveurs POP ou IMAP, transitent en clair (sans cryptage) sur le réseau et peuvent être aisément capturés, permettant ainsi l'usurpation du compte e-mail.

De même, le contenu des messages est également transmis en clair sur le réseau.

4. ACCES FACILE A DES LISTES D'E-MAIL

Le développement du « spam » a généré une activité importante de création et de vente de listes, voire de CD d'adresses e-mail, plus ou moins valides, qui sont utilisés pour l'envoi massif de messages, soit commerciaux, soit plus « brutaux » comme des virus par exemple.

De plus, de nombreuses listes ne sont pas protégées et permettent l'énumération des destinataires.

Enfin, les données intrinsèques liées aux domaines de l'Internet, et qui sont contenues dans les DNS[3] décrivent les propriétaires des noms de domaines avec leurs coordonnées, ainsi que les contacts administratifs ou techniques.

Ces données sont rarement protégées laissant une place importante à leur récupération pour le « spam » par simple interrogation des DNS.

De même, les e-mails inscrits dans les pages web des sites d'entreprises par exemple sont facile à récupérer par des robots qui obtiennent ainsi de nombreuses coordonnées.

[3] DNS : Domain Name System

5. LES « CHAINES » DE MESSAGES SOCIAUX

Parmi les « spams » courants, les chaînes de messages *« à retransmettre à tous vos contacts » ;* on peut distinguer celles qui font croire à une menace importante à diffuser rapidement, et celles liées à un problème général ou une personne en détresse.

Il est systématiquement demandé à l'utilisateur de diffuser ce messages à l'ensemble de ses contacts, en incluant parfois en copie l'émetteur d'origine. L'envoi de ces adresses e-mail, qui transitent en clair (sans cryptage) sur Internet permet aux spammeurs de récupérer celles-ci en les validant, soit parce que l'ordinateur de l'utilisateur est déjà « investi » par les spammeurs dans le cadre d'un réseau de « botnets »[4], soit par une écoute attentive du réseau ou une interception des communications entre l'utilisateur et son serveur de messagerie.

6. LES ASPECTS DE LA CONFIDENTIALITE

La plupart des opérateurs de confiance et les ISPs rejettent le plus souvent l'idée de la confidentialité, sous le prétexte d'inutilité, de complexité ou de risque de perte des données sur le long terme.

En fait, la crainte de voir la plupart des utilisateurs transférer des messages indéchiffrables (ou presque) semble être à l'origine de ce rejet.

[4] Botnet : réseau de PC manipulés à distance par des spammeurs à l'insu de leurs propriétaires

Pour ces raisons, les opérateurs de confiance ont émis des certificats de signature électronique de classe III qui ne peuvent pas être utilisés pour crypter un message.

Les arguments liés à la durée de vie du certificat et à la complexité de gestion d'un mécanisme de récupération de clés appelé « key recovery » par le tiers de confiance ont été les bases de ce choix.

Pourtant, le besoin de confidentialité est important et nécessaire afin de pouvoir transférer des documents en toute confiance, que ce soit pour des données sensibles comme de nouveaux développements de brevets, des courriers d'avocats, des chiffres confidentiels…

La question de la confidentialité deviendra une question encore plus épineuse avec le développement du « cloud computing » dans lequel les données pourront être situées n'importe où dans le monde sans protection quand à leur accès par des tiers.

L'objectif de la confidentialité n'est pas la dissimulation, mais la sécurité des informations contenues dans des messages électroniques ou dans des documents.

7. LA FACILITE DE MODIFICATION DES MESSAGES

Aucune fonction de protection du contenu des messages n'étant en place dans les protocoles et la traçabilité des messages étant déficiente, il est particulièrement aisé d'altérer le contenu d'un message pour en modifier le sens.

Particulièrement lors des transferts de messages, n'importe quel utilisateur se rend compte qu'il peut modifier le contenu du message qu'il a reçu et qu'il veut transférer.

En utilisant la fonction de son client de messagerie « transférer » (forward), le message d'origine est affiché et modifiable aussi simplement que la rédaction d'un message.

Dès lors, il suffit de l'imprimer une fois modifié pour prétendre avoir reçu un message contenant ces éléments modifiés.

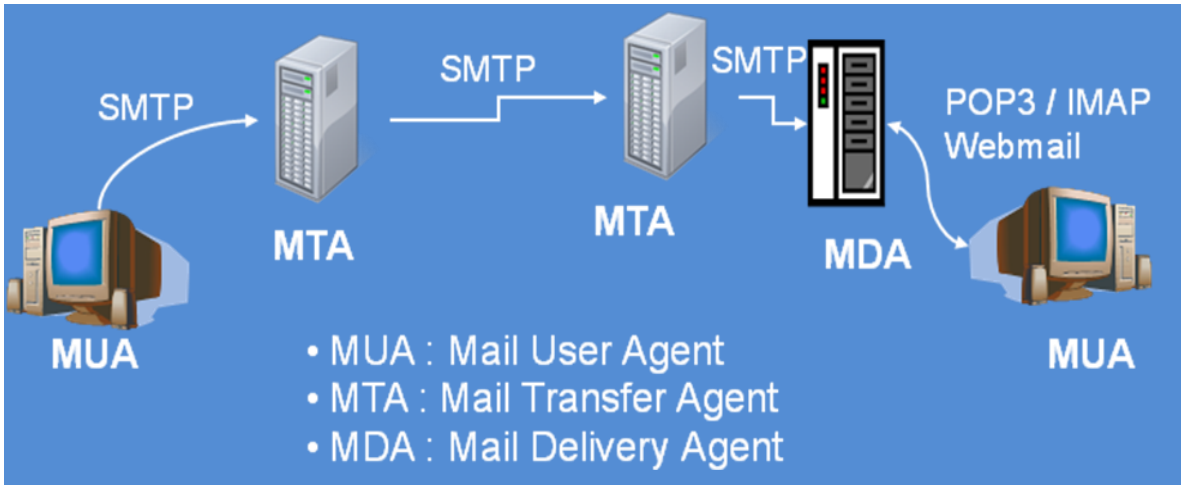
Ce phénomène se rencontre souvent lors de la présentation en justice d'impression de messages prétendument reçus dont le contenu a été volontairement altéré.

III. Analyse de l'existant

1. PRINCIPE GENERAL DE LA MESSAGERIE INTERNET

1. Schéma général

Le mécanisme schématique du fonctionnement de la messagerie Internet est le suivant :



L'utilisateur dont l'adresse est « emetteur@chezmoi.com », sur son poste de travail, va rédiger un message pour un utilisateur «user@domaineloin.com».

Son logiciel de messagerie va transmettre ce message, en général, au serveur d'envoi (SMTP) de son domaine : « smtp.chezmoi.com ».

Ce serveur va analyser la demande et en particulier l'adresse du destinataire «user@domaineloin.com». Il va donc essayer de contacter le domaine

« domaineloin.com » et lui demander par une requête DNS quelle est l'adresse de son serveur de messagerie, qu'il trouve dans le contenu de la table DNS (champ MX).

Le DNS est le mécanisme hiérarchique au niveau mondial qui permet d'associer un nom « lisible » et compréhensible par un humain, à l'adresse IP de l'ordinateur en question (ex : 213.186.33.5 en IPV4)

Selon le cas, il peut avoir une réponse lui donnant directement l'adresse du serveur dont il s'agit, ou le nom d'un serveur « relay » qui, lui, pourra accéder au serveur de destination sollicité.

Le serveur « smtp.chezmoi.com » va donc établir une session avec soit le serveur de destination « smtp.domaineloin.com », soit avec un serveur « relay », et lui transmettre le message en y ajoutant des informations de 'transit' telles que son adresse IP et les date et heure de passage (« timestamp »).

Le serveur « smtp.domaineloin.com » recevant la demande, va vérifier s'il existe bien un utilisateur nommé « user » dans son domaine.
Si tel n'est pas le cas, il renvoie un message d'erreur à l'émetteur du message, *via* le serveur SMTP émetteur (smtp.chezmoi.com).

> *REMARQUE : il convient de noter que dans le cas où l'adresse de l'émetteur n'existe pas ou est erronée, le message risque de générer des allers-retours entre les deux serveurs SMTP en boucle. C'est ce qu'on appelle le « bouncing »*

Si l'utilisateur existe bien dans son domaine, le serveur SMTP « smtp.domaineloin.com » va transmettre au MDA local (Message Delivery Agent) le message afin qu'il soit stocké dans l'attente de sa consultation par le destinataire.

Enfin, le destinataire, à partir de son poste de travail et de son client de messagerie (MUA), va se connecter sur le MDA *via* un des protocoles les plus courants, POP3 (Post Office Protocol) ou IMAP4 (Internet Message Access Protocol) pour aller télécharger ou consulter le message qui lui a été envoyé.

Pour ce faire, il s'authentifie auprès du serveur (MDA) avec son e-mail (ou nom d'utilisateur) et son mot de passe, avant de pouvoir consulter son message.

Il est important de noter que le protocole SMTP a été conçu pour un environnement connecté de bout en bout, ce qui implique le fonctionnement permanent des serveurs de messagerie SMTP et l'existence des MDA (serveurs POP ou IMAP) pour accepter en permanence les messages alors que l'utilisateur n'est pas toujours connecté.

2. Structure des messages

Les messages électroniques sont composés de trois parties essentielles, savoir :

- Les en-têtes (headers) des messages comportant les informations de l'émetteur, du(es) destinataire(s), l'objet et les traces de l'envoi et du transit du message.
- Le corps du message lui-même, celui-ci pouvant être rédigé en texte brut ou en HTML avec images et autres fonctions.
- Les pièces attachées (les fichiers divers joints au message lui-même).

Le fonctionnement historique de l'Internet et en particulier des serveurs SMTP, implique que seuls des caractères ASCII sur 7 bits sont acceptés par ces serveurs.

En conséquence, les envois sont transcodés de l'ASCII 8bits vers l'ASCII 7bits afin de pouvoir être transmis correctement, selon différents modes de codage de caractères, ce qui peut générer des erreurs de transcriptions lors de la réception (caractères accentués, étrangers..).

Ceci devient d'autant plus flagrant que l'utilisation de l'Unicode peut entraîner des confusions supplémentaires qui sont largement utilisées par les « spammeurs ».

Les fichiers attachés sont généralement codés avec les extensions MIME (Multipurpose Internet Mail Extensions : RFC 2045 et 2046)[5].

Le codage des caractères fait parfois appel également à ces extensions.

Les références des protocoles de messagerie principaux sont les suivants :

- SMTP : RFC 5321
- POP3 : RFC 1939
- IMAP4 : RFC 3401
- Structure des messages : RFC 2822
- En-têtes des messages : RFC 5322

[5] RFC : Request For Comment : c.f. chapitre 5.a

3. Types de messageries

On peut séparer le fonctionnement des messageries selon trois modes distincts :

- **Messageries d'entreprises :**

  Dans le cas des messageries d'entreprises, le(s) serveur(s) de messagerie SMTP sont hébergés au sein de l'entreprise elle-même et accessibles en permanence.

  Souvent, celles-ci sont du type Microsoft Exchange ou Lotus Domino, et moins fréquemment des produits comme Zimbra.

  Avec ce type d'architecture, les fichiers de messagerie sont conservés sur le serveur et les utilisateurs y accèdent directement via leur client de messagerie (Outlook, Notes…).
  Parfois, des copies de ces données sont répliquées sur le poste de travail, mais ce n'est pas systématique.

- **Client de messagerie utilisateur final :**

  Cette catégorie représente la plus grande partie des messageries utilisées à ce jour.

  Le serveur de messagerie (MDA) est hébergé chez le fournisseur d'accès ou de gestion du domaine (chez l'ISP).

  L'utilisateur utilise le client de messagerie de son poste de travail pour accéder *via* les protocoles POP ou IMAP au contenu de sa messagerie stockée sur les serveurs (MDA).

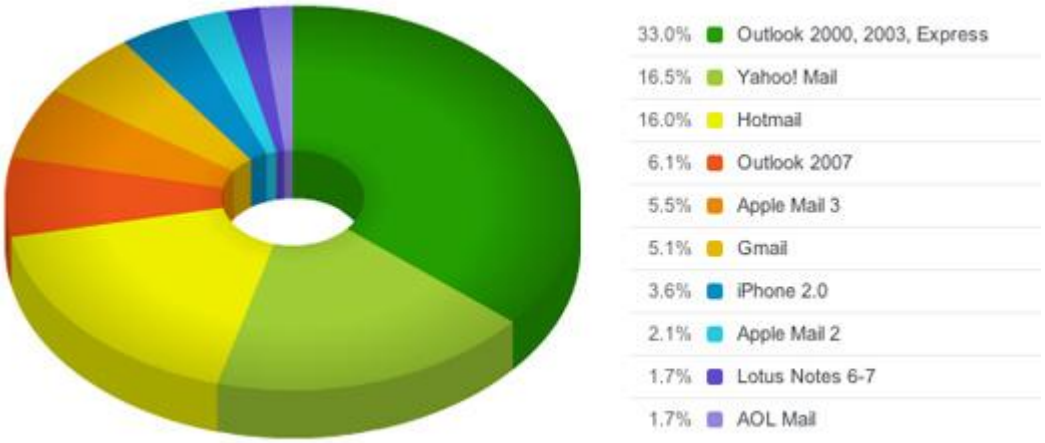
Pour IMAP, les messages sont simplement consultés localement mais conservés sur les serveurs, alors qu'avec le protocole POP, les messages sont téléchargés sur le poste de travail du client.

- **Fonctionnement distant (Webmail) :**

Enfin, de plus en plus fréquemment, l'usage de serveurs distants accessibles essentiellement par un navigateur Internet, est employé par les particuliers.

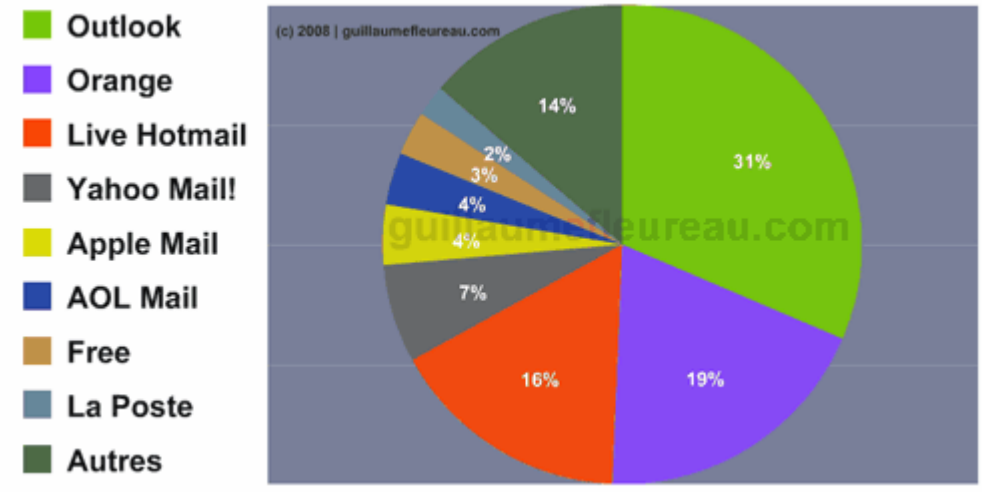
Dans ce cas, comme pour le protocole IMAP, les messages sont consultés à distance avec le navigateur et restent stockés sur le serveur en question (ex : Gmail, Yahoo, Hotmail…)

Répartition mondiale des types de messageries :



Source : Campaign Monitor

Répartition en France :



Source : Guillaume Fleureau

Dans les deux cas, on constate une forte majorité d'utilisateurs des logiciels Microsoft Outlook qui sont livrés soit avec le système Windows (Outlook express) soit avec Microsoft Office (Outlook 2000 à 2010).

La part des « webmails » devient de plus en plus importante.

2. LES PROTOCOLES DE MESSAGERIE

1. Présentation générale

▪ Qu'est-ce qu'un RFC ?

Les RFC (Request For Comment) représentent la manière de normaliser les protocoles de l'Internet.

Aux débuts de l'Internet, lorsque les protocoles d'échange ont été conçus, il a été nécessaire de les valider, puis de les améliorer au cours du temps. C'est ainsi que sont nés les RFC pour gérer les propositions et demandes d'amélioration des protocoles déjà en cours d'utilisation.

Ceci a permis une accélération sensible de la mise en œuvre puis du développement de ces protocoles, et en particulier de TCP/IP.

Par comparaison, en Europe, le principe a été différent :

Des ingénieurs ont commencé par rédiger des normes d'échange (X25) pendant plusieurs années afin de les normaliser.

Puis, il a été demandé aux constructeurs de concevoir et fabriquer des matériels respectant ces normes, ce qui a encore demandé plusieurs années.

Enfin, il a fallu mettre en œuvre tous ces matériels et normes autour d'un réseau physique de communication.

Ces lourdeurs et le temps passé à normaliser ont eu pour conséquences tout d'abord un retard du démarrage des réseaux de communication entre ordinateurs, et d'autre part un coût très élevé des systèmes de connexion proposés.

Ceci, entre autres, explique le rapide développement de TCP/IP aux dépens de X25.

Comme nous l'avons vu au chapitre 3. Les principaux protocoles de messagerie que nous analyserons plus en détail sont les suivants :

- SMTP : RFC 5321 (anciennement 2821)
- POP3 : RFC 1939
- IMAP4 : RFC 3401
- Structure des messages : RFC 2822
- En-têtes des messages : RFC 5322

▪ Le protocole SMTP

Simple Mail Transfer Protocol.

Ainsi que son nom l'indique est un protocole très simplifié pour la transmission de messages en texte clair envoyés sur un lien permanent, c'est-à-dire entre plusieurs machines toujours présentes et actives sur Internet.

Son ancienneté explique sa simplicité, car aux débuts d'Internet on ne se préoccupait pas des problèmes de sécurité et encore moins des aspects liés à l'identification des émetteurs de messages. Le « spam » n'existait pas et les utilisateurs recevaient peu de messages.

Sa simplicité explique également son succès et la vitesse de sa mise en œuvre dans l'Internet, car il est facile à implémenter dans un programme, et on trouve encore actuellement certains petits programmes le mettant en œuvre de façon simpliste, mais respectant l'ensemble des règles décrites dans son RFC initial 821, mis à jour par le RFC 2821 puis par le RFC 5321.

Tous les MTAs (Message Transfer Agent) respectent cette RFC et le protocole SMTP.

De fait, ils en héritent les faiblesses intrinsèques ainsi que les éventuels problèmes d'implémentation,

Plus précisément, le mécanisme de ces serveurs est basé sur un ou plusieurs fichiers de configuration au format texte (en clair) qui peut être facilement modifié dès lors que l'on peut avoir accès à l'ordinateur sur lequel il se trouve.

En fait, ces fichiers de configuration sont si « touffus » et complexes qu'il est peu recommandé d'éditer directement ceux-ci, en particulier le fichier de configuration principal du logiciel le plus répandu : « sendmail.cf ».

Pour éviter les éventuels problèmes de configuration, un macroprocesseur (m4) pour modifier le fichier « sendmail.cf ».

▪ Les protocoles POP et IMAP

Les MDAs (**Message Delivery Agent**) sont le plus souvent des serveurs supportant les protocoles POP[6] et IMAP[7] afin de permettre aux utilisateurs de se connecter de façon épisodique pour relever leurs messages de la même manière que nous allons chercher notre courrier papier dans notre boite aux lettres.

[6] Post Office Protocol
[7] Internet Message Access Protocol

Historiquement, les premiers MDAs stockaient simplement les fichiers de façon individuelle dans des sous-dossiers du dossier personnel du destinataire, sur le disque dur de destination, qui était en permanence connecté.

L'évolution de ces systèmes et la diffusion de l'Internet auprès des PME et des particuliers a développé la connectivité intermittente et, donc, l'usage des serveurs POP et IMAP.

Les mécanismes de ces deux protocoles sont sensiblement différents.

Initialement, seul le protocole POP existait et sa simplicité résidait dans le peu de fonctions qu'il comportait, essentiellement la connexion avec *« login et mot de passe »* et le téléchargement séquentiel des messages conservés sur le serveur.

Puis, le développement de l'Internet a nécessité :

D'une part des extensions successives au protocole POP :

> POP3 qui comporte des extensions de type UIDL pour identifier les messages ; POP-AUTH pour authentifier l'utilisateur, ou encore l'établissement d'un lien SSL entre le client et le serveur.

Et d'autre part, la mise en œuvre d'un nouveau protocole, IMAP (actuellement IMAP4), pour gérer de façon plus étendue les fonctionnalités de la messagerie, en particulier le stockage permanent des messages sur le serveur et leur consultation distante.

Le principe était lié à la faible vitesse des liens de transmission existants alors pour les particuliers associée à l'augmentation du nombre et de la taille des messages transmis.

Ce protocole évitait ainsi le téléchargement systématique des messages sur le poste de travail et la gestion distante de ces messages.

Les évolutions actuelles en termes de vitesse des liens et les aspects liés à la confidentialité des données réduisent, de mon point de vue, sensiblement l'intérêt de ce protocole.

Je considère que l'amélioration des fonctions sur le poste de travail au sein du logiciel « client » de messagerie est préférable, dès lors que l'on sécurise les fonctionnalités du protocole POP3.

En effet, la conservation sur le serveur distant de l'ensemble des messages peut générer de nombreux risques :

- Difficulté de sauvegarde des données et risque de perte d'informations,
- Disponibilité des messages lorsque l'ordinateur n'est pas connecté à Internet,
- Plus grande facilité d'accès pour des tiers malveillants,
- Risques liés à la confidentialité des données présentes sur le serveur : Au moins les administrateurs du serveur peuvent avoir accès au contenu des messages.

3. QU'EST-CE QUE LE «SPAM[8]» ?

1. Les origines et objectifs du «spam»

Le «spam» est un terme anglo-saxon issu d'une caricature liée à un produit en conserve de mauvaise qualité !

Il est issu d'un show des « Monty Python » « Flying Circus » dans lequel, une serveuse ne propose que des produits accompagnés de « SPAM » et qui est repris en boucle par les personnes présentes (132 fois pendant le sketch), y compris dans le générique[9].

Ce terme a été utilisé pour qualifier les messages électroniques indésirables reçus dans nos boîtes aux lettres électroniques.

Dans le monde dématérialisé de l'Internet ces envois non sollicités prennent des proportions gigantesques, et rapidement les boîtes de messagerie sont saturées de messages indésirables, car l'envoi d'un e-mail est peu coûteux et facile à réaliser en nombre.

Les logiciels d'envoi massif de messages sont nombreux, et comportent des fonctionnalités avancées permettant notamment des envois anonymes et l'utilisation de nombreuses ressources à travers le monde.

Même si le premier envoi reconnu comme un « spam » date de 1978, ce phénomène reste relativement récent et s'est amplifié avec une rapidité fulgurante dès la fin des années 90.

[8] Le mot SPAM en majuscule fait référence et est protégé par la société SPAM qui produit des conserves. Seule l'écriture en minuscules est acceptée pour définir le courriel électronique non sollicité.
[9] http://www.youtube.com/watch?v=anwy2MPT5RE

Dans le milieu des années 90, on comptait encore peu d'internautes individuels, aussi, les envois massifs de messages n'avaient que peu d'efficacité et se trouvaient de fait moins nombreux.

En revanche l'utilisation de l'Internet s'étant développé rapidement, le modèle économique du « spam » est devenu rentable et lui-même a crût aussi vite, voire même plus vite.

2. Constat d'augmentation du fléau

Le «spam» a réellement débuté vers le milieu des années 90 et a commencé à être considéré comme un fléau à partir de 1997.

Dès ce moment, de nombreux outils et méthodes pour contrer le «spam» ont été mise en place pour essayer de « filtrer » les messages, une fois reçus dans nos « boîtes aux lettres ».

La proportion du «spam» dans le total des échanges par e-mails a subi une croissance exponentielle depuis le milieu des années 90.

A titre d'exemple, on peut partir de l'année 2001, pendant laquelle environ 40% des emails mondiaux étaient considérés comme non-sollicités, laissant ainsi près de 60% de messages « utiles ».

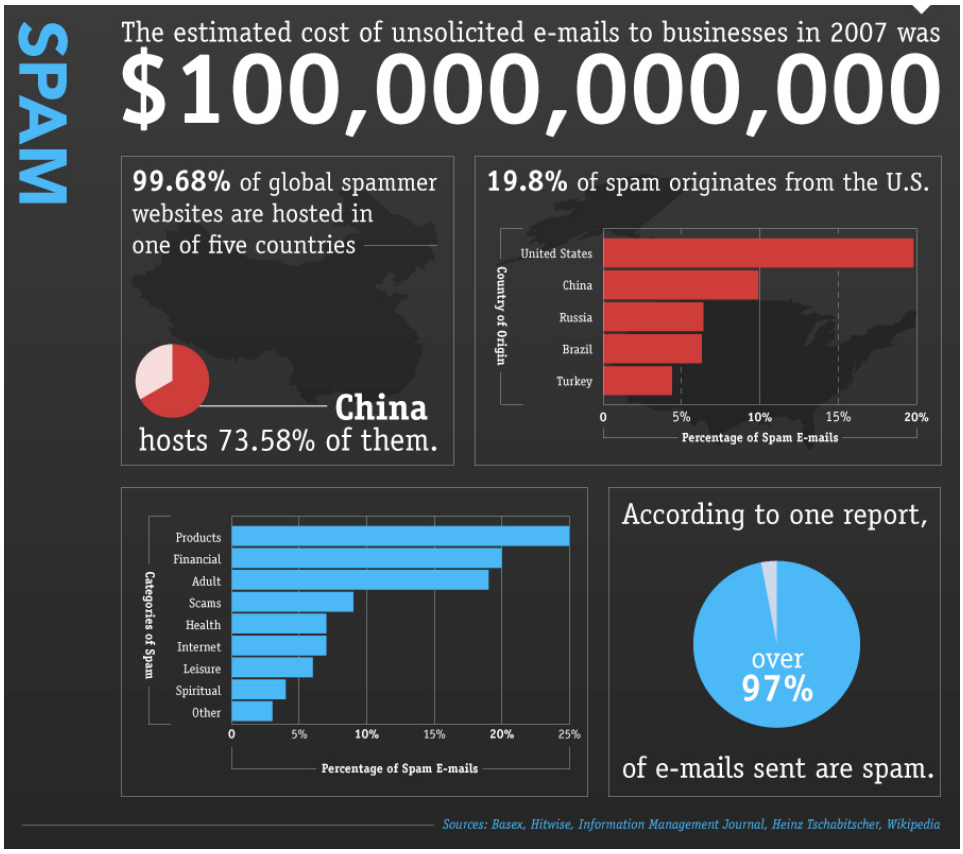
Or, uniquement en France, en 2005, 80% des messages échangés étaient vus comme du «spam», et ce taux est passé à 95% en 2006, **réduisant la part des messages « utiles » à 5% !**

3. Les conséquences sur les coûts

Sur les bases précitées, en prenant des proportions conservatrices, si un utilisateur passe 5mn par jour (soit environ 30 h/an) pour trier les emails indésirables de ceux qui peuvent être utiles, cela correspond à 57 Milliards d'heures de perdues par an, soit environ **24 millions d'années/hommes perdues par an…**

Cette rapide et simple évaluation explique clairement la notion de « fléau » et le phénomène de saturation ressenti par les utilisateurs.

En 2007, la perte estimée pour les entreprises, liée à la gestion du « spam » était de 100 Milliards de $, ainsi que le présente le schéma suivant :



4. Traitements actuels des effets

L'essentiel des outils actuels vendus sur le marché ont pour vocation de **traiter uniquement les effets du « spam » et non les causes.**

En général, deux niveaux de filtres existent sur les messageries actuelles :

Un premier filtre fonctionne sur les serveurs de messagerie gérés par les opérateurs Internet, ce qui permet de réduire le nombre de ces messages non sollicités dans la boîte aux lettres de l'utilisateur final.

Le deuxième niveau est réalisé, selon la décision de l'utilisateur final de se protéger ou non, sur son poste de travail afin de filtrer et trier les messages arrivant dans sa boîte aux lettres selon certaines règles automatisées ou qui lui sont propres.

Les conséquences directes de ce fonctionnement sont les suivantes :

a. Une charge de fonctionnement importante sur le poste de travail de l'utilisateur pouvant même saturer la mémoire de celui-ci dans certains cas.

b. Une course et escalade technique entre les « spammeurs » et les vendeurs de solutions anti-spam, qui mettent en place de nouveaux moyens techniques, à l'image de cette escalade qui existe entre les cambrioleurs et les fabricants de serrures.

Enfin, la mise en œuvre et l'utilisation de ces outils ne réduisent en rien la charge subie par les moyens d'accès Internet (les « tuyaux ») car le trafic global généré par les « spams » n'est pas réduit dans la mesure où les traitements s'effectuent à l'arrivée des messages en bout de chaîne.

5. Les causes du «spam»

Parmi les causes du « spam », et au-delà du modèle économique évoqué plus avant, il convient de mettre en avant quelques causes essentielles qui font l'objet de la présente thèse :

- **Absence d'identification de l'émetteur**

Les protocoles de messagerie utilisés couramment au sein de l'Internet, ne valident pas l'identité de l'émetteur d'un message électronique.

Des outils d'entreprise tels que Microsoft Exchange ou Lotus Domino traitent ces aspects, mais ces fonctions ne sont pas « propagées » de serveur en serveur au sein de la sphère Internet.

- **Pas de vérification de l'existence d'une adresse e-mail**

De même, les serveurs SMTP utilisés pour les envois de messages, ne réalisent pas de vérification de l'existence de l'adresse e-mail de l'émetteur et acceptent même des envois de messages sans adresse e-mail source.

- **Pas de sécurisation systématique au niveau du protocole**

Les protocoles utilisés, et qui seront détaillés plus avant, possèdent certaines fonctions limitées de sécurisation des échanges entre le poste de travail et les serveurs.

Néanmoins leur utilisation n'est pas systématique et, de ce fait, la plupart des échanges se font de manière non sécurisée à plusieurs niveaux.

4. EVOLUTION DU «SPAM»

1. Classification

La classification du « spam » est une tâche ardue, tant les exemples se multiplient et les idées des spammeurs évoluent.

Néanmoins, certaines personnes ont cherché à définir des catégories typiques de « spam » en fonction des objectifs de ceux-ci :

- La sollicitation de vente directe de produits,
- La publicité pour de nouveaux produits ou des promotions,
- La promotion de sites webs,
- Le développement commercial sous couvert d'informations,
- Les « Chaînes » sociales,
- Les escroqueries diverses telles que la fraude 4-1-9 (nigériane)

Et bien d'autres catégories de « spam » dits malveillants comme l'installation de virus ou de backdoors sur les postes de travail.

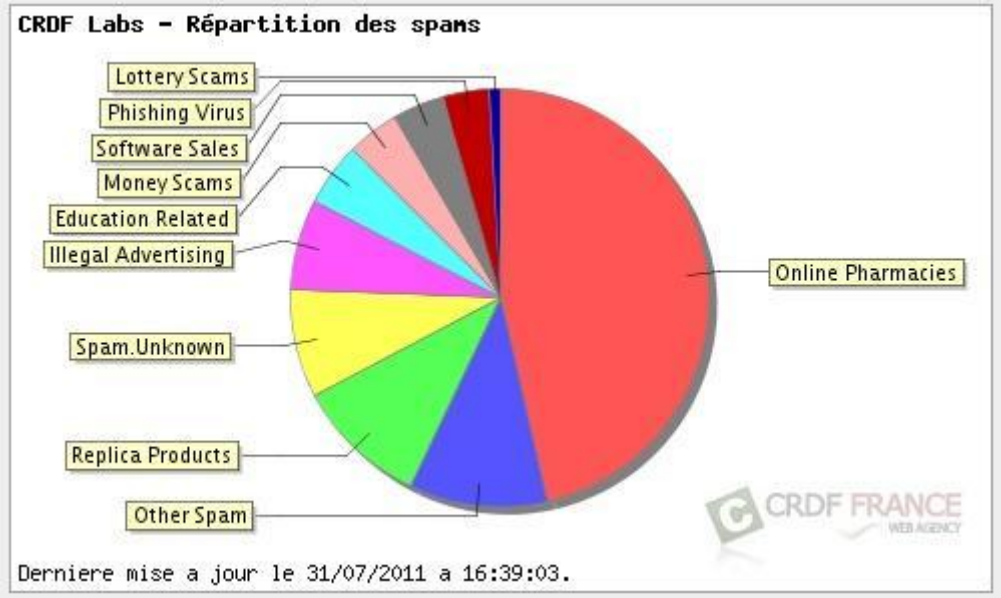
Une classification simple entre les « spams » commerciaux et ceux qualifiés de « malveillants » peut être représentée de la façon suivante :

Types de spams commerciaux en 2008	**%**
Sexe	22,5
Commerce courant	21,3
Contrefaçons	7,5
Conseils boursiers	2,2
Crédit	1,5
Sites de pub	1,1
Gagner de l'argent	0,5
Sous-total	**56,6**

Types de spam malveillant en 2008	**%**
Pharmacies	10,7
Jeux d'argent en ligne	9,1
Ventes de logiciel OEM	8,3
Sites Web malveillants	7,2
Outils de spammer	4,3
Programmes malveillants	0,9
Sous-total	**40,5**

Source : « sécurité internet » : http://eservice.free.fr/

Informations sur la répartition des catégories de « spams » :



Et en fonction de leur provenance :

2. Economie du «spam»

Les marchés liés au « spam » peuvent être séparés en trois domaines :

- Les ventes directes ou indirectes proposées par le « spam »
- Les outils nécessaires pour le « spam »
- Les outils « anti-spam »

Economiquement, le premier marché, représenté par l'envoi de messages de masse est plutôt rentable pour les sociétés qui utilisent cette méthode de publicité, car le coût d'achat de listes d'emails valides reste peu onéreux et le coût d'envoi est minime.

L'achat d'une base de données d'e-mails valides pour réaliser des envois massifs varie entre 140 € pour 5 Millions d'emails dont beaucoup ne sont pas valides, à des prix pouvant atteindre plusieurs milliers d'euros pour 100.000 emails très qualifiés, incluant adresse, fonction et téléphone par exemple.

Le spammeur va plutôt s'orienter vers des bases de données à bas prix, puis utilisera des réseaux de « botnets » (cf paragraphe c.) peu onéreux, entre 60 et 150 € par jour, et qui peuvent facilement envoyer plusieurs dizaines de millions d'emails par jour.

Ceci représente donc un coût total de moins de 500 € pour envoyer plus de 10 Millions de messages.

En conséquence, même si une bonne partie des « spams » ne sont pas consultés par les utilisateurs, soit parce qu'ils sont filtrés ou détruits par l'utilisateur, soit parce que le contenu n'intéresse pas l'internaute, il reste environ 0,001% d'utilisateurs qui cliquent sur les liens proposés et accèdent à ces sites pour acheter les produits ou services proposés.

Ce marché est estimé à environ **250 millions d'internautes qui achètent les produits ou services du « spam ».**

Sur cette base de 0,001% de retour positif généré par ces envois de « spam », on peut estimer les ventes journalières à près 2,5 millions, soit plus de **900 millions de ventes par an.**

En prenant une valeur basse par vente, soit 10 € en moyenne, cela représente **un chiffre d'affaires, généré par le « spam » pour les annonceurs et vendeurs, de plus de 9 Milliards d'euros par an.**

Le second marché lié aux *« outils pour spammeurs »* représente environ **500 Millions d'euros par** an entre les achats d'adresses et l'utilisation des serveurs d'envoi massifs (botnets : réseau de zombies…).

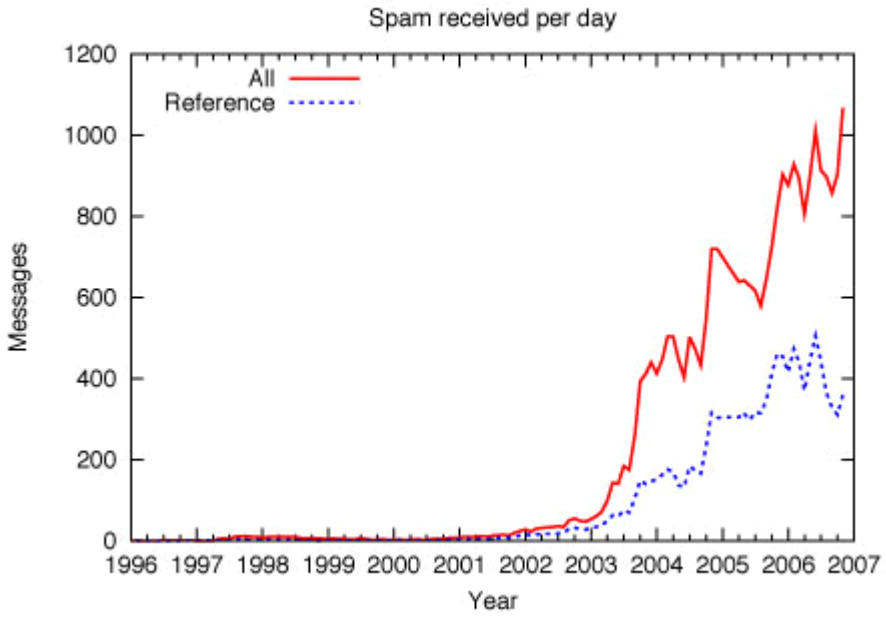
Enfin, le marché des **outils anti-spam** se situe également dans la zone des **500 Millions d'euros par an**.

La somme de ces marchés représente donc une **économie mondiale de l'ordre de 10 Milliards d'euros par an !**

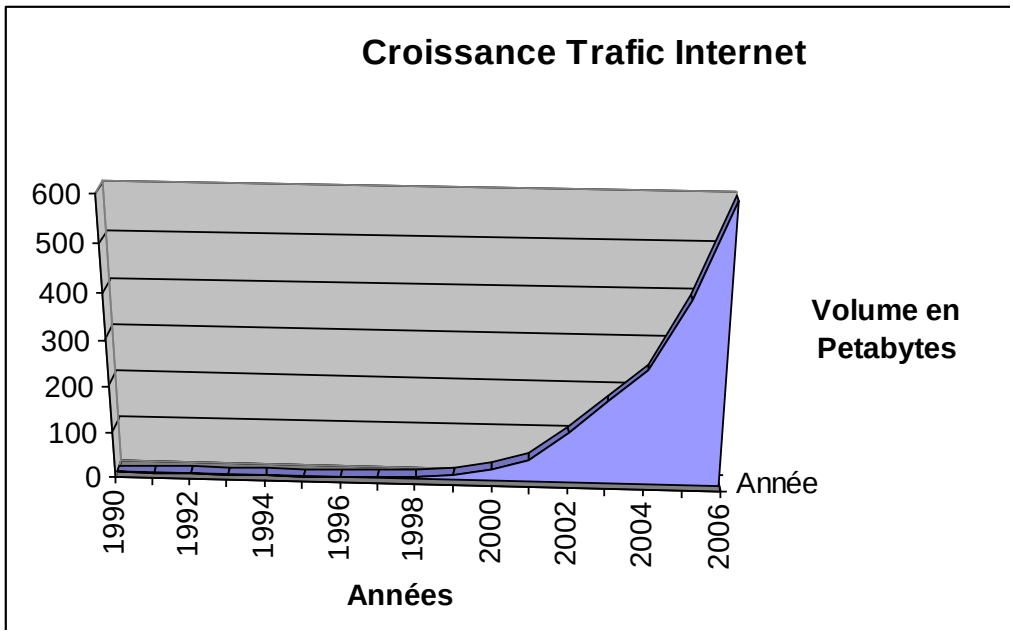
On comprend alors aisément que ce système soit très rentable pour les spammeurs et ne peut qu'augmenter.

3. Augmentation du trafic indésirable

Parallèlement à la croissance du nombre des utilisateurs de l'Internet, l'augmentation du « spam » a été plus rapide encore :



Source : http://spamnation.info/stats/



Données calculées

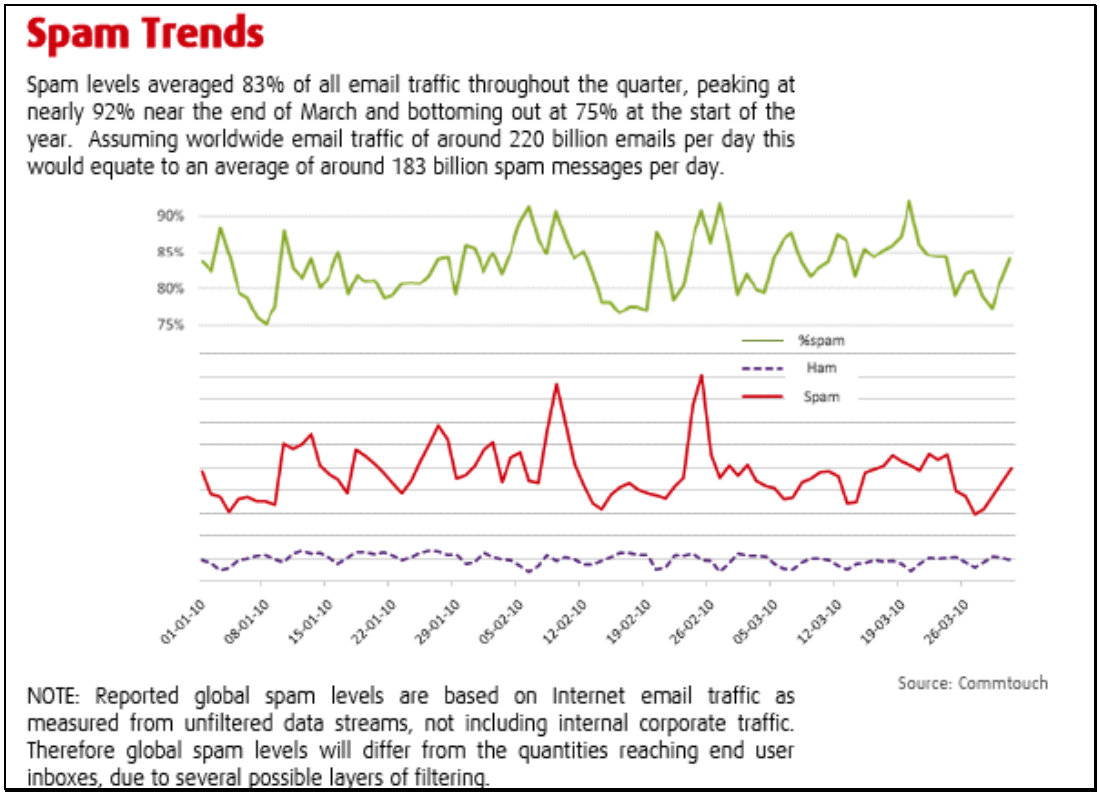
Le taux de « spam » est monté jusqu'à 97% en 2009 puis est redescendu épisodiquement à 75% en 2010 grâce à l'arrêt de plusieurs sites très gros émetteurs de « spam » :

> *En Novembre 2009, Oleg Nikolaenko, un Russe de 23 ans est arrêté pour un spam d'envergure mondiale. Il dirigeait le réseau de Botnets Mega-D, un réseau d'ordinateurs servant à envoyer des spams et qui à son apogée était responsable d'un tiers du volume mondial de spam, avec plus de 10 Milliards de pourriels par jour !*
>
> *En Février 2010, Microsoft fait fermer le Botnet Waledac. Ce réseau d'ordinateurs infectés était utilisé par des hackers pour envoyer du spam et représentait l'un des plus gros botnets basé aux Etats-Unis. On estime que Waledac a infecté plusieurs centaines de milliers de machines dans le monde.*
>
> *Octobre 2010, Igor A. Gusev, le roi des spammers est en fuite. Réputé comme l'un des plus gros spammers au monde et résidant à Moscou, cet homme, âgé de 31 dirigeait SpamIt.com, une entreprise embauchant des spammeurs pour faire la publicité de pharmacies en ligne. Coïncidence ou non, le nombre de spams a chuté de près de 20% depuis sa disparition.*
>
> *Il faut noter que même si les systèmes de commande de ces réseaux ont été désactivés, ces démantèlements n'ont pas nettoyé les ordinateurs infectés qui restent toujours sensibles aux botnets et autres malwares.*

(Source « secuserve »)

Néanmoins, le risque de remontée rapide est toujours présent et expliqué par l'économie du « spam » ainsi que nous l'avons vu au chapitre précédent.

Précisions sur le trafic du « spam » en 2010 :



Actuellement, il transite environ 295 Milliards d'e-mails par jour dans le monde, dont environ 81% de « spam », soit <u>plus de 235 Milliards de « spams » par jour</u> !

L'ensemble de ces données doit être ramené au nombre d'utilisateurs de la messagerie électronique qui se situe, à l'heure actuelle à environ **2,3 Milliards au niveau mondial.**

Autrement dit, en moyenne, chaque utilisateur reçoit environ 100 « spams » par jour…

5. LES METHODES STATISTIQUES APPLICABLES

Le domaine statistique est vaste, et il ne convient pas de le décrire ici.

Néanmoins, compte tenu du nombre et de la taille des échanges dématérialisés, nous vivons au cœur de cette matière.
Aussi, certains aspects de cette science sont nécessaires, tant pour analyser ce qui se passe que pour aider à combattre le « spam ».

En premier lieu, les fonctions de BAYES sur les probabilités relatives sont utilisées pour décider si le contenu d'un message est ou n'est pas du « spam ».

Ensuite, l'analyse stochastique et les processus de Markov sont utiles pour la définition de modèles et de prévisions.

Enfin, le « goodness of fit » partant de la vectorisation du modèle pour le comparer aux résultats mesurés nous apportera une information significative sur la pertinence du modèle.

6. SOLUTIONS ACTUELLES « ANTI-SPAM »

1. Traitement des effets

Ainsi que je l'ai rappelé précédemment, les solutions actuelles utilisées contre le « spam » traitent principalement les effets, c'est-à-dire le tri des messages une fois reçus dans la boîte aux lettres.

Néanmoins, quelques rares solutions existent pour essayer de filtrer les messages en amont de leur réception.

2. Classification des méthodes de filtrage

Les différents modes de filtrage existant actuellement sont appliquées soit au niveau du fournisseur d'accès (ISP), soit au niveau du poste de travail de l'utilisateur et parfois aux deux niveaux.

L'objectif étant, selon le type de méthode, de repérer les adresses IP ou les domaines qui sont à l'origine de l'envoi des « spams » du côté ISP, ou d'analyser le contenu des messages afin de déterminer s'ils appartiennent à cette catégorie ou non, puis de les stocker dans un dossier de type « courrier indésirable » côté utilisateur.

Certaines fonctionnalités étendues au niveau de la messagerie sont tentées pour identifier plus facilement le domaine de l'émetteur du message.

On peut ainsi classifier ces méthodes de la façon suivante :

- DNSBL (DNS Black Listing) : C'est une méthode qui consiste à identifier les adresses IP des serveurs de messagerie des spammeurs et de fournir cette liste en ligne *via* un service de requête DNS spécifique (ex : RBL).

- Black List : cela correspond au terme générique qui consiste à rejeter des adresses spécifiquement identifiées comme étant à l'origine du « spam ». Plus particulièrement ce mécanisme est implémenté au niveau de la plupart des clients de messagerie pour trier les messages en provenance d'adresses e-mail spécifiques.

- White list : C'est le mécanisme opposé du précédent qui revient à spécifier dans une liste d'adresse e-mails celles dans lesquelles l'utilisateur a confiance explicitement.

- « Hard White List » : En allant plus loin dans le même principe, plusieurs services en ligne proposent de valider systématiquement les nouvelles adresses e-mail que l'on peut recevoir en demandant explicitement à l'émetteur de se connecter sur un lien Web et de valider un « captcha » (séquence de caractères graphiques lisible par un humain), afin de valider son adresse d'envoi de message.

- Filtrage sur le contenu : cette méthode est utilisée soit au niveau de l'ISP, soit sur le poste de travail de l'utilisateur, grâce aux logiciels dits « anti-spam » en analysant le contenu du message, en particulier à l'aide de filtres « bayesiens », aussi bien au niveau des en-têtes que du corps du message lui-même. Ensuite le tri est effectué, avec toutes les limites de cette analyse et les détournements réalisés par les spammeurs.

- SPF (Sender Policy Framework) (RFC 4408) : Projet qui consiste à créer une sorte de « white list » des serveurs de messagerie envoyant des e-mails en testant les enregistrements DNS du domaine de l'émetteur.

- Sender ID (RFC 4406) : Fonctionne sur le même principe que SPF, mais va un peu plus loin en testant les champs « From » « Sender », « resent from » qui sont vérifiés lors d'une session SMTP.

- DKIM (Domain Key Identified Mail) : est une solution intégrant un mécanisme de clés asymétriques qui permet d'ajouter la signature du serveur de messagerie d'envoi associé à un domaine, ce qui en principe, identifie clairement le serveur de messagerie émetteur et évite le « domaine spoofing ».

3. Limites des outils

Ces méthodes et outils possèdent de nombreuses limites, même lorsqu'ils sont combinés entre eux.

Le DNS Black List provoque fréquemment des « surblocages » de sites émettant des messages normaux, car souvent les serveurs utilisés par les spammeurs sont aussi utilisés à des fins normales d'envoi à des listes de diffusion par exemple, sans compter les serveurs de messagerie utilisés à l'insu de leur propriétaire par les spammeurs.

Par exemple, les listes de diffusion de la FNTC sont parfois vues par certains opérateurs comme Orange comme venant d'un serveur de SPAM, car ces listes sont gérées par la société OVH, gros hébergeur, qui peut héberger des spammeurs. La conséquence de ce sur-filtrage est l'absence de réception de messages importants de l'association par ses membres.

La gestion de 'black lists' repose sur la gestion et la mise à jour de celles-ci, sachant que les émetteurs de « spams » changent fréquemment d'adresses e-mails, voire utilisent de fausses adresses, et leurs serveurs d'envoi de messages changent souvent pour éviter, justement, les inscriptions dans ces listes.

Les 'white lists' ne définissent que des émetteurs « surs » mais n'empêchent pas la réception des « spams ».

Les « hard white lists » sont plus contraignantes et filtrent effectivement beaucoup de spammeurs, mais également la réception de messages importants de confirmation d'enregistrement sur un site, d'envoi de mots de passe par ses fournisseurs, et surtout rebute d'éventuels clients qui n'iront pas s'enregistrer sur le site pour valider leur « existence ».

Le filtrage de contenu possède une certaine efficacité sur les messages déjà reçus dans sa boîte aux lettres et permet de filtrer probablement 50% des

« spams ». Néanmoins, il existe de nombreux faux positifs qui obligent à vérifier le dossier des messages ainsi triés avant de les supprimer, ce qui fait perdre beaucoup de temps.
Même si les techniques de filtrage utilisent des méthodes comme les filtres Bayesiens qui permettent une analyse statistique pointue, celles-ci sont détournées par les « spammeurs » qui améliorent également leurs techniques.

SPF et Sender ID permettent de réduire légèrement les « spams » arrivant sur le serveur final en refusant certains serveurs émetteur. Mais ces mécanismes sont trop peu utilisés et n'empêchent pas les émissions de nombreux spammeurs.

DKIM a l'avantage d'éviter la dissimulation derrière un domaine qui n'est pas le sien à l'aide de clés asymétriques et d'un serveur gérant ces identificateurs, mais il dépend justement de ces serveurs d'identification, de ceux qui génèrent les clés et, possède les mêmes limites que SPF et Sender ID. Par contre, son approche est intéressante et introduit un des éléments de sécurisation de la messagerie présentée dans cette thèse.

4. Conséquences sur le trafic Internet

L'ensemble de ces solutions mise en place depuis des années a permis de réduire sensiblement le taux de « spams » reçus dans les boîtes aux lettres des utilisateurs.

On estime que **20% des « spams » continuent à passer les barrières** mise en place par l'ensemble de ces outils et méthodes.

Ce qui représente encore 50 Milliards de « spam » par jour !

IV. Les aspects juridiques du « spam »

Dans son rapport du 14 octobre 1999 intitulé « Le publipostage électronique et la protection des données », la CNIL donnait une définition du « spam », faisait un état des techniques utilisées par les spammeurs et des solutions proposées par les différents acteurs, et rappelait celle retenue Parlement européen à l'époque le « op-out »[10].

En France, afin de lutter contre le « spam », le réseau « Signal Spam » par exemple, a été créé en 2005, sous la forme d'une association de loi 1901 qui regroupe la plupart des organisations françaises concernées par la lutte contre le spam, qu'il s'agisse des pouvoirs publics ou des professionnels de l'Internet, et a pour objet de fédérer les efforts de tous pour lutter contre le fléau du spam.

Son action repose sur le signalement par les internautes d'un « spam ». Les informations transmises sont relayées vers les acteurs concernés afin qu'une action soit menée pour agir à la source de l'envoi (enquêtes judiciaires des services de Police, contrôle de la CNIL chez les entreprises dans le cas de campagnes abusives, identification par les fournisseurs d'accès des clients dont les ordinateurs sont infectés et utilisés dans les réseaux d'envoi (botnet).

En île de France, et plus exclusivement à Paris et petite couronne (départements 92-93-94), il existe une brigade d'enquête sur les fraudes aux technologies de l'information la « BEFTI ». Cette brigade composée de 28 agents dont 18 enquêteurs traite de nombreuses affaires telles le sabotage des données informatiques ; l'attaque d'un système informatique ;

[10] Opt-out : possibilité de décocher une option pré-sélectionnée

l'espionnage ; le blocage de sites Internet ; la contrefaçon de logiciels ou de bases de données et les atteintes aux droits d'auteur sur Internet et n'a, de ce fait, pas la possibilité de lutter efficacement contre le « spam ».

Pourtant dès 1997, la question du « spam » était abordée et pas moins de quatre directives contradictoires proposaient une solution avant l'adoption de la LCEN (Loi pour la confiance dans l'économie numérique du 21 juin 2004).

En effet, la directive n° 97/7/CE du 20 mai 1997 relative à la protection des consommateurs en matière de contrats à distance consacrait le système du « opt-out ».

Puis la directive n° 97/67/CE du 15 décembre 1997 concernant le traitement des données à caractère personnel et la protection de la vie privée dans le secteur des télécommunications à distance laissait les États libres de choisir quelle formule adopter pour lutter.

Celle du 8 juin 2000 n° 2000/31/CE relative à certains aspects du commerce électronique était plus en faveur du opt-out malgré quelques ambiguïtés et enfin la directive n° 2002/58/CE du 12 juillet 2002 concernant le traitement des données à caractère personnel et la protection de la vie privée dans le secteur des communications électroniques était quant à elle franchement favorable au système du « opt-in »[11], en ces termes :

« Il importe de protéger les abonnés contre toute violation de leur vie privée par des communications non sollicitées effectuées à des fins de prospection directe, en particulier au moyen d'automates d'appel, de télécopies et de courriers électroniques, y compris les messages courts (SMS). Si ces formes de communications commerciales non sollicitées peuvent être relativement faciles et peu onéreuses à envoyer, elles peuvent, en revanche imposer une charge et/ou un coût à leur destinataire. En outre, dans certains cas, leur volume peut poser un problème pour les réseaux de communications électroniques et les équipements terminaux. S'agissant de ces formes de communications non sollicitées effectuées à des fins de prospection directe, ***il est justifié d'exiger de l'expéditeur qu'il ait obtenu le consentement préalable du destinataire avant de les lui envoyer****... »*

[11] Opt-in : possibilité de cocher une option non sélectionnée

Finalement, les dispositions de l'article 1135 du Code civil (« Les conventions obligent non seulement à ce qui y est exprimé, mais encore à toutes les suites que l'équité, l'usage ou la loi donnent à l'obligation d'après sa nature ») se sont révélées être un outil très efficace puisque deux décisions prononcées par le Tribunal de grande instance de Rochefort sur Mer (jugement du 28 février 2001) et par le Président du Tribunal de grande instance de Paris (ordonnance du 15 janvier 2002), ont condamné des spammeurs en utilisant ce visa.

L'utilisation du spamming est alors jugée « contraire aux usages de l'Internet » et justifie la « résiliation du contrat d'accès au réseau ».

Enfin la question était également envisagée par les articles 323-1 et 323-2 (issus de la loi sur les atteintes aux systèmes de traitement informatisé des données du 5 janvier 1988 dite loi Godfrain) du Code pénal prohibant l'accès et le maintien dans un système de traitement automatisé de données ainsi que l'entrave au fonctionnement d'un tel système.

Pourtant, la LCEN est venue renforcer la qualification pénale du « spam ».

L'apport de la LCEN (Loi pour la confiance dans l'économie numérique du 21 juin 2004) est une option de compromis.

Par principe le « spam » est condamné (l'article 22 de la dite loi que l'on retrouvera dans le Code des postes et communications électroniques codifié à l'article 34-5 et dans le Code de la consommation à l'article L 121-20-5) en ces termes :

« *Est interdite la prospection directe au moyen d'un automate d'appel, d'un télécopieur ou d'un courrier électronique utilisant, sous quelque forme que ce soit, les coordonnées d'une personne physique qui n'a pas exprimé son consentement préalable à recevoir des prospections directes par ce moyen.*

Pour l'application du présent article, on entend par consentement toute manifestation de volonté libre, spécifique et informée par laquelle une personne accepte que des données à caractère personnel la concernant soient utilisées à fin de prospection directe.

Constitue une prospection directe l'envoi de tout message destiné à promouvoir, directement ou indirectement, des biens, des services ou l'image d'une personne vendant des biens ou fournissant des services.

Toutefois, la prospection directe par courrier électronique est autorisée si les coordonnées du destinataire ont été recueillies directement auprès de lui, dans le respect des dispositions de la loi n° 78-17 du 6 janvier 1978 relative à l'informatique, aux fichiers et aux libertés, à l'occasion d'une vente ou d'une prestation de services, si la prospection directe concerne des produits ou services analogues fournis par la même personne physique ou morale, et si le destinataire se voit offrir, de manière expresse et dénuée d'ambiguïté, la possibilité de s'opposer, sans frais, hormis ceux liés à la transmission du refus, et de manière simple, à l'utilisation de ses coordonnées lorsque celles-ci sont recueillies et chaque fois qu'un courrier électronique de prospection lui est adressé.

Dans tous les cas, il est interdit d'émettre, à des fins de prospection directe, des messages au moyen d'automates d'appel, télécopieurs et courriers électroniques, sans indiquer de coordonnées valables auxquelles le destinataire puisse utilement transmettre une demande tendant à obtenir que ces communications cessent sans frais autres que ceux liés à la transmission de celle-ci. Il est également interdit de dissimuler l'identité de la personne pour le compte de laquelle la communication est émise et de mentionner un objet sans rapport avec la prestation ou le service proposé.

La Commission nationale de l'informatique et des libertés veille, pour ce qui concerne la prospection directe utilisant les coordonnées d'une personne physique, au respect des dispositions du présent article en utilisant les compétences qui lui sont reconnues par la loi n° 78-17 du 6 janvier 1978 précitée. A cette fin, elle peut notamment recevoir, par tous moyens, les plaintes relatives aux infractions aux dispositions du présent article.

Les infractions aux dispositions du présent article sont recherchées et constatées dans les conditions fixées par les premier, troisième et quatrième alinéas de l'article L. 450-1 et les articles L. 450-2, L. 450-3, L. 450-4, L. 450-7, L. 450-8, L. 470-1 et L. 470-5 du code de commerce.

Un décret en Conseil d'Etat précise en tant que de besoin les conditions d'application du présent article, notamment eu égard aux différentes technologies utilisées ».

Clairement, l'option « opt-in » est retenue par la LCEN et en cela, la France rejoint ses homologues européens allemands, autrichiens, danois, finlandais, italiens et belges.

Mais compromis car selon ce même article 22 :

« Toutefois, la prospection directe par courrier électronique est autorisée si les coordonnées du destinataire ont été recueillies directement auprès de lui, dans le respect des dispositions de la loi n° 78-17 du 6 janvier 1978 relative à l'informatique, aux fichiers et aux libertés, à l'occasion d'une vente ou d'une prestation de services, si la prospection directe concerne des produits ou services analogues fournis par la même personne physique ou morale, et si le destinataire se voit offrir, de manière expresse et dénuée d'ambiguïté, la possibilité de s'opposer, sans frais, hormis ceux liés à la transmission du refus, et de manière simple, à l'utilisation de ses coordonnées lorsque celles-ci sont recueillies et chaque fois qu'un courrier électronique de prospection lui est adressé ».

Le compromis requiert trois conditions cumulatives :

- la collecte des données est légale si elle est directe et conforme à la loi relative à l'informatique aux fichiers et aux libertés,

- la prospection possible si elle concerne des produits ou services analogues,

- le destinataire se voit offrir la possibilité de refuser le procédé.

Par ailleurs, la Commission nationale de l'informatique et des libertés (Cnil) est venue préciser qu'un formulaire doit être adressé à chaque personne afin de recueillir son consentement relatif à l'envoi de messages commerciaux, et ce même document doit permettre de refuser toute utilisation ultérieure de ces données.
En effet, les dispositions des articles 226-18 et 226-18-1 du Code pénal interdit la récupération des adresses électroniques sur le Web de manière automatique ainsi que le commerce de celles-ci en ces termes :

Article 226-18 du Code Pénal :

« Le fait de collecter des données à caractère personnel par un moyen frauduleux, déloyal ou illicite est puni de cinq ans d'emprisonnement et de 300000 Euros d'amende ».

Article 226-18-1 :

« Le fait de procéder à un traitement de données à caractère personnel concernant une personne physique malgré l'opposition de cette personne, lorsque ce traitement répond à des fins de prospection, notamment commerciale, ou lorsque cette opposition est fondée sur des motifs légitimes, est puni de cinq ans d'emprisonnement et de 300 000 Euros d'amende ».

Il a été rappelé que la directive européenne 2002/58/CE rend obligatoire le consentement préalable des destinataires de messages électroniques publicitaires, selon le principe de « l'opt-in ».

Pourtant, dès 1995, la Directive 95/46/CE du Parlement européen et du Conseil, du 24 octobre évoquait déjà ces questions et précisait dans ses articles :

Article 2

*« h) **"consentement de la personne concernée"** : toute manifestation de volonté, libre, spécifique et informée par laquelle la personne concernée accepte que des données à caractère personnel la concernant fassent l'objet d'un traitement ».*

Article 6

« Les États membres prévoient que les données à caractère personnel doivent être :

a) traitées loyalement et licitement
*b) collectées pour des finalités déterminées, explicites et légitimes, et **ne pas être traitées ultérieurement de manière incompatible avec ces finalité ».***

Article 7

« Les États membres prévoient que le traitement de données à caractère personnel ne peut être effectué que si :

*a) **la personne concernée a indubitablement donné son consentement ».***

Article 14

« Droit d'opposition de la personne concernée.

Les États membres reconnaissent à la personne concernée le droit d'être informée avant que des données à caractère personnel ne soient pour la première fois communiquées à des tiers ou utilisées pour le compte de tiers à des fins de prospection et de se voir expressément offrir le droit de s'opposer, gratuitement, à ladite communication ou utilisation ».

Article 17

« Sécurité des traitements

Les États membres prévoient que le responsable du traitement doit mettre en œuvre les mesures techniques et d'organisation appropriées pour protéger les données à caractère personnel contre la destruction accidentelle ou illicite, la perte accidentelle, l'altération, la diffusion ou l'accès non autorisés, notamment lorsque le traitement comporte des transmissions de données dans un réseau, ainsi que contre toute autre forme de traitement illicite ».

La loi dite Godefrain du 5 janvier 1986 relative aux fraudes informatiques dans son chapitre trois mentionnait également :

Article 462-2

« Quiconque, frauduleusement, aura accédé ou se sera maintenu dans tout ou partie d'un système de traitement automatisé de données sera puni d'un emprisonnement d'un mois à un an et d'une amende de 2.000F à 50.000F ou de l'une de ces deux peines. Lorsqu'il en sera résulté soit la suppression ou la modification de données contenues dans le système, soit une altération du fonctionnement de ce système, l'emprisonnement sera de deux mois à deux ans et l'amende de 10.000F à 100.000F ».

Article 462-3

« Quiconque aura intentionnellement et au mépris des droits d'autrui, entravé ou faussé le fonctionnement d'un système de traitement automatisé de données sera puni d'un emprisonnement de trois mois à trois ans et d'une amende de 10.000F à 100.000F ou de l'une de ces deux peines ».

Article 462-4

« Quiconque aura, intentionnellement et au mépris des droits d'autrui, directement ou indirectement, introduit des données dans un système de traitement automatique ou supprimé ou modifié les données qu'il contient ou leurs modes de traitement ou de transmission, sera puni d'un emprisonnement de trois mois à trois ans et d'une amende de 2.000F à 500.000F ou de l'une de ces deux peines ».

Ces dispositions sont applicables au « spam » qui peut, d'une part, être porteur de virus qui sont des intrusions dans un système de traitement automatisé de données, et d'autre part, peut venir perturber le système de l'utilisateur, par la consommation de bande passante ou l'emploi de ressources de l'ordinateur.

L'arsenal juridique existant pouvait déjà être utilisé pour sanctionner les spammeurs.
Pourtant, la LCEN (Loi pour la Confiance dans l'Economie Numérique : **Loi n°2004-575 du 21 juin 2004**) dispose, dans son article 20 :

Article 20

« Toute publicité, sous quelque forme que ce soit, accessible par un service de communication au public en ligne, doit pouvoir être clairement identifiée comme telle. Elle doit rendre ***clairement identifiable la personne physique ou morale pour le compte de laquelle elle est réalisée.***

L'alinéa précédent s'applique sans préjudice des dispositions réprimant les pratiques commerciales trompeuses prévues à l'article L. 121-1 du code de la consommation ».

Et dans son article 22 :

*« III. - Sans préjudice des articles L. 33-4-1 du code des postes et télécommunications et L. 121-20-5 du code de la consommation tels qu'ils résultent des I et II du présent article, le consentement des personnes dont les coordonnées ont été recueillies avant la publication de la présente loi, dans les conditions prévues par la loi n° 78-17 du 6 janvier 1978 relative à l'informatique, aux fichiers et aux libertés, à l'utilisation de celles-ci à fin de prospection directe peut être sollicité, par voie de courrier électronique, pendant les six mois suivant la publication de la présente loi. A l'expiration de ce délai****, ces personnes sont présumées avoir refusé l'utilisation ultérieure de leurs coordonnées personnelles à fin de prospection directe si elles n'ont pas manifesté expressément leur consentement à celle-ci ».***

L'article 20 de cette Loi est important, car il valide la nécessité d'identification, si ce n'est l'authentification de l'émetteur du message publicitaire, or dans la plupart des cas liés au « spam », cette identification n'est pas réalisée, et c'est un des piliers de la présente thèse.

Le consentement de son côté nécessite la mise en place de « l'opt-in » qui permet à l'utilisateur de valider une demande d'envoi de messages, ce qui, bien entendu, n'est pas réalisé par la plupart des spammeurs.

Enfin, il a été rappelé que la CNIL agissait, elle aussi, activement contre le « spam » :

« La CNIL et Signal spam, partenaires dans la lutte contre le spam

30 octobre 2007

Le mardi 30 octobre, Alex Türk, Président de la CNIL, et Dominique Roux, Président de Signal spam, ont signé une convention de partenariat définissant les modalités d'intervention des deux institutions dans la lutte contre le spam qui demeure un des problèmes majeurs d'internet et mine la confiance dans l'économie numérique.

La CNIL pionnière contre le spam

Le spam s'appuie sur la collecte illicite d'adresses électroniques de particuliers et menace la sécurité des réseaux. C'est donc une pratique contraire à la loi « informatique et libertés ».

La CNIL mène depuis plusieurs années une politique active de lutte contre le spam, qu'il s'agisse de l'application effective de la législation anti-spam, de l'adoption de codes de bonne conduite par les professionnels, ou encore du développement d'une forte coopération internationale.

La loi pour la confiance dans l'économie numérique du 21 juin 2004 est d'ailleurs venue rappeler la compétence particulière de la CNIL dans la lutte contre l'envoi de spams à des personnes physiques et la possibilité, pour elle, de recevoir par tous moyens les plaintes relatives aux infractions aux règles de prospection électronique.

Soucieuse d'appréhender de façon concrète le phénomène du spam, la CNIL avait créé en 2002 un dispositif appelé « boite à spam » invitant les internautes à transférer par courrier électronique leurs messages non sollicités. Cette opération a trouvé immédiatement un grand écho auprès du public avec la réception de plus de 300.000 messages en trois mois ».

Le « spam » est non seulement un délit prévu et réprimé en tant que tel par les dispositions du Code pénal, mais est aussi une faute civile permettant d'engager la responsabilité de son auteur (spammeur) afin d'obtenir réparation (dommages-intérêts).

V. Généralités sur la signature électronique

L'évolution de la dématérialisation des documents a rendu nécessaire la mise en place d'un moyen permettant d'identifier un document et d'authentifier son auteur. La signature électronique, au même titre que la signature manuscrite vis-à-vis d'un document papier, permet d'authentifier un document dématérialisé.

1. ASPECTS LEGISLATIFS

La Directive Européenne du 13 décembre 1999, qui a été transposée en droit interne le 13 Mars 2000, pose le principe de l'équivalence entre l'écrit électronique et l'écrit papier, et a été codifiée de façon précise dans le Code Civil dans les articles suivants :

« Art. 1316-1. - L'écrit sous forme électronique est admis en preuve au même titre que l'écrit sur support papier, sous réserve que puisse être dûment identifiée la personne dont il émane et qu'il soit établi et conservé dans des conditions de nature à en garantir l'intégrité. »

Et :

« Art. 1316-3. - L'écrit sur support électronique a la même force probante que l'écrit sur support papier. »

Et :

« Art. 1316-4. - La signature nécessaire à la perfection d'un acte juridique identifie celui qui l'appose. Elle manifeste le consentement des parties aux obligations qui découlent de cet acte. Quand elle est apposée par un officier public, elle confère l'authenticité à l'acte.

Lorsqu'elle est électronique, elle consiste en l'usage d'un procédé fiable d'identification garantissant son lien avec l'acte auquel elle s'attache. La fiabilité de ce procédé est présumée, jusqu'à preuve contraire, lorsque la

signature électronique est créée, l'identité du signataire assurée et l'intégrité de l'acte garantie, dans des conditions fixées par décret en Conseil d'Etat. »

Il convient de noter également les articles 1 et 2 du décret du 30 Mars 2001 qui précisent :

« Article 1

Au sens du présent décret, on entend par :

1. Signature électronique : une donnée qui résulte de l'usage d'un procédé répondant aux conditions définies à la première phrase du second alinéa de l'article 1316-4 du code civil ;

2. Signature électronique sécurisée : une signature électronique qui satisfait, en outre, aux exigences suivantes :

- être propre au signataire ;

- être créée par des moyens que le signataire puisse garder sous son contrôle exclusif ;

- garantir avec l'acte auquel elle s'attache un lien tel que toute modification ultérieure de l'acte soit détectable ;

3. Signataire : toute personne physique, agissant pour son propre compte ou pour celui de la personne physique ou morale qu'elle représente, qui met en œuvre un dispositif de création de signature électronique ;

4. Données de création de signature électronique : les éléments propres au signataire, tels que des clés cryptographiques privées, utilisés par lui pour créer une signature électronique ;

5. Dispositif de création de signature électronique : un matériel ou un logiciel destiné à mettre en application les données de création de signature électronique ;

6. Dispositif sécurisé de création de signature électronique : un dispositif de création de signature électronique qui satisfait aux exigences définies au I de l'article 3 ;

7. Données de vérification de signature électronique : les éléments, tels que des clés cryptographiques publiques, utilisés pour vérifier la signature électronique ;

8. Dispositif de vérification de signature électronique : un matériel ou un logiciel destiné à mettre en application les données de vérification de signature électronique ;

9. Certificat électronique : un document sous forme électronique attestant du lien entre les données de vérification de signature électronique et un signataire ;

10. Certificat électronique qualifié : un certificat électronique répondant aux exigences définies à l'article 6 ;

11. Prestataire de services de certification électronique : toute personne qui délivre des certificats électroniques ou fournit d'autres services en matière de signature électronique ;

12. Qualification des prestataires de services de certification électronique : l'acte par lequel un tiers, dit organisme de qualification, atteste qu'un prestataire de services de certification électronique fournit des prestations conformes à des exigences particulières de qualité.

Article 2

La fiabilité d'un procédé de signature électronique est présumée jusqu'à preuve contraire lorsque ce procédé met en œuvre une signature électronique sécurisée, établie grâce à un dispositif sécurisé de création de signature électronique et que la vérification de cette signature repose sur l'utilisation d'un certificat électronique qualifié ».

L'utilisation d'une signature électronique « normale », représentant *« l'usage d'un procédé répondant aux conditions définies à la première phrase du second alinéa de l'article 1316-4 du code civil »* et le respect des termes de l'Article 2.2 du présent décret, est suffisante pour valider un message électronique ou un document.

L'utilisation d'une **signature électronique basée sur un certificat électronique qualifié** *(de classe IIII)* a pour effet complémentaire la **présomption de fiabilité**, autrement dit le renversement de la charge de la preuve.

En complément des textes relatifs à la signature électronique proprement dite, il convient de mettre en avant également les textes visant **les moyens de cryptographie**, car ceux-ci sont étroitement liés à l'utilisation de la signature électronique, ainsi qu'aux aspects de la confidentialité qui sont importants dans le domaine de la confiance.

La LCEN (Loi pour la Confiance dans l'Economie Numérique : **Loi n°2004-575 du 21 juin 2004**) précise les mécanismes autorisés dans le domaine de la cryptologie, définissant ceux nécessitant une déclaration ou une autorisation :

Article 29

« On entend par moyen de cryptologie tout matériel ou logiciel conçu ou modifié pour transformer des données, qu'il s'agisse d'informations ou de signaux, à l'aide de conventions secrètes ou pour réaliser l'opération inverse avec ou sans convention secrète. ***Ces moyens de cryptologie ont principalement pour objet de garantir la sécurité du stockage ou de la transmission de données, en permettant d'assurer leur confidentialité, leur authentification ou le contrôle de leur intégrité.***

On entend par prestation de cryptologie toute opération visant à la mise en œuvre, pour le compte d'autrui, de moyens de cryptologie. »

Article 30

« I. ***- L'utilisation des moyens de cryptologie est libre.***

II. - La fourniture, le transfert depuis ou vers un Etat membre de la Communauté européenne, l'importation et l'exportation des moyens de cryptologie assurant exclusivement des fonctions d'authentification ou de contrôle d'intégrité sont libres.

III. - La fourniture, le transfert depuis un Etat membre de la Communauté européenne ou l'importation d'un moyen de cryptologie n'assurant pas exclusivement des fonctions d'authentification ou de contrôle d'intégrité sont soumis à une déclaration préalable auprès du Premier ministre, sauf dans les cas prévus au b du présent III. Le fournisseur ou la personne procédant au transfert ou à l'importation tiennent à la disposition du Premier ministre une description des caractéristiques techniques de ce moyen de cryptologie, ainsi que le code source des logiciels utilisés. Un décret en Conseil d'Etat fixe :

a) Les conditions dans lesquelles sont souscrites ces déclarations, les conditions et les délais dans lesquels le Premier ministre peut demander communication des caractéristiques du moyen, ainsi que la nature de ces caractéristiques ;

b) Les catégories de moyens dont les caractéristiques techniques ou les conditions d'utilisation sont telles que, au regard des intérêts de la défense nationale et de la sécurité intérieure ou extérieure de l'Etat, leur fourniture, leur transfert depuis un Etat membre de la Communauté européenne ou leur importation peuvent être dispensés de toute formalité préalable.

IV. - Le transfert vers un Etat membre de la Communauté européenne et l'exportation d'un moyen de cryptologie n'assurant pas exclusivement des fonctions d'authentification ou de contrôle d'intégrité sont soumis à autorisation du Premier ministre, sauf dans les cas prévus au b du présent IV.

Un décret en Conseil d'Etat fixe :

a) Les conditions dans lesquelles sont souscrites les demandes d'autorisation ainsi que les délais dans lesquels le Premier ministre statue sur ces demandes ;

b) Les catégories de moyens dont les caractéristiques techniques ou les conditions d'utilisation sont telles que, au regard des intérêts de la défense nationale et de la sécurité intérieure ou extérieure de l'Etat, leur transfert vers un Etat membre de la Communauté européenne ou leur exportation peuvent être soit soumis au régime déclaratif et aux obligations d'information prévus au III, soit dispensés de toute formalité préalable. »

Article 31

« I. - La fourniture de prestations de cryptologie doit être déclarée auprès du Premier ministre. Un décret en Conseil d'Etat définit les conditions dans lesquelles est effectuée cette déclaration et peut prévoir des exceptions à cette obligation pour les prestations dont les caractéristiques techniques ou les conditions de fourniture sont telles que, au regard des intérêts de la défense nationale et de la sécurité intérieure ou extérieure de l'Etat, cette fourniture peut être dispensée de toute formalité préalable.

II. - Les personnes exerçant cette activité sont assujetties au secret professionnel, dans les conditions prévues aux articles 226-13 et 226-14 du code pénal. »

Article 32

*« Sauf à démontrer qu'elles n'ont commis aucune faute intentionnelle ou négligence**, les personnes fournissant des prestations de cryptologie à des fins de confidentialité sont responsables au titre de ces prestations**, nonobstant toute stipulation contractuelle contraire, du préjudice causé aux personnes leur confiant la gestion de leurs conventions secrètes en cas d'atteinte à l'intégrité, à la confidentialité ou à la disponibilité des données transformées à l'aide de ces conventions. »*

Article 33

« Sauf à démontrer qu'ils n'ont commis aucune faute intentionnelle ou négligence, les prestataires de services de certification électronique sont responsables du préjudice causé aux personnes qui se sont fiées raisonnablement aux certificats présentés par eux comme qualifiés dans chacun des cas suivants :

1° Les informations contenues dans le certificat, à la date de sa délivrance, étaient inexactes ;

2° Les données prescrites pour que le certificat puisse être regardé comme qualifié étaient incomplètes ;

3° La délivrance du certificat n'a pas donné lieu à la vérification que le signataire détient la convention privée correspondant à la convention publique de ce certificat ;

4° Les prestataires n'ont pas, le cas échéant, fait procéder à l'enregistrement de la révocation du certificat et tenu cette information à la disposition des tiers.

Les prestataires ne sont pas responsables du préjudice causé par un usage du certificat dépassant les limites fixées à son utilisation ou à la valeur des transactions pour lesquelles il peut être utilisé, à condition que ces limites figurent dans le certificat et soient accessibles aux utilisateurs.

Ils doivent justifier d'une garantie financière suffisante, spécialement affectée au paiement des sommes qu'ils pourraient devoir aux personnes s'étant fiées raisonnablement aux certificats qualifiés qu'ils délivrent, ou d'une assurance garantissant les conséquences pécuniaires de leur responsabilité civile professionnelle. »

Ces textes montrent clairement l'importance de la cryptographie dans l'ensemble du processus de la confiance, tant pour la signature électronique que pour la confidentialité des données ou pour la vérification de leur intégrité.

L'usage de ces procédés est libre, mais la fourniture de ce type de services est soumise à déclaration et porte une responsabilité claire imposant la mise en œuvre de mécanismes de récupération des clés.

L'ensemble de ces textes concernant la signature électronique et la cryptographie a donc fourni une équivalence entre le document papier et le document électronique et précisé les moyens d'y parvenir en toute confiance.

D'autre part, l'article 1316-4 a précisé la notion de signature électronique, ainsi que de signature simple, qui consiste en la *« manifestation du consentement »*.

Il est important de noter la différence entre **le certificat électronique** qui est un outil technique, et **la signature électronique** qui est *la « manifestation du consentement »* par l'application et l'utilisation du certificat électronique. Cette différence est clairement expliqué dans l'article 1 du décret du 30 Mars 2001.

En effet, la confusion est couramment réalisée entre les deux notions, et l'on appelle souvent « signature électronique » l'outil technique qu'est le certificat.

L'usage, depuis dix ans environ, consiste à valoriser uniquement la signature électronique « sécurisée » basée sur un certificat « qualifié », au détriment de la signature électronique simple ou sécurisée avec un certificat non qualifié.

Pourtant, aux termes de l'Article 1316-4 du Code Civil, la signature électronique « simple » est suffisante dès lors que l'usage d'un procédé fiable d'identification garantit son lien avec l'acte auquel elle s'attache.

Les trois points principaux spécifiant une « signature électronique sécurisée basée sur un certificat qualifié » sont liés à l'émission dudit certificat :

- La certification de l'Autorité de Certification qui émet les certificats
- L'authentification de l'identité « civile » du porteur du certificat par l'Autorité d'Enregistrement
- La fourniture du certificat sur un support externe sécurisé

De nombreux décrets et Lois insistent sur la sécurisation de la signature électronique, ainsi que sur les moyens d'accès aux réseaux de la Justice par exemple.
De même et de plus en plus fréquemment, les réponses aux Appels d'Offres requièrent l'usage d'une signature électronique sécurisée et qualifiée par le RGS (Référentiel Général de Sécurité)

La volonté générale d'identifier de façon **certaine** le signataire a ainsi promu l'utilisation d'un certificat de signature électronique délivré par une Autorité de Confiance, et remis, en face à face, par une Autorité d'Enregistrement, sur un support sécurisé de type carte à puce (généralement sous la forme d'une clé USB).

Pour autant, **les fonctions techniques de signature et de cryptage éventuel, ainsi que l'authentification d'une adresse e-mail sont communes aux deux catégories de certificats.**

Or, ce sont justement ces fonctions qui peuvent être utilisées pour accompagner le développement de la confiance dans la messagerie électronique ainsi que nous le verrons plus loin, et l'usage d'une signature électronique « sécurisée » n'est pas indispensable dans ce cas.

2. FONCTIONNEMENT D'UNE ICP[12]

Ce fonctionnement a permis de développer des architectures de confiance, basées techniquement sur une **Infrastructure à Clés Publiques** (ICP ou PKI pour Public Key Infrastructure en Anglais).

Ces infrastructures sont basées sur des technologies de cryptographie asymétrique, c'est-à-dire l'utilisation de couples de clés (clé privée et clé publique).

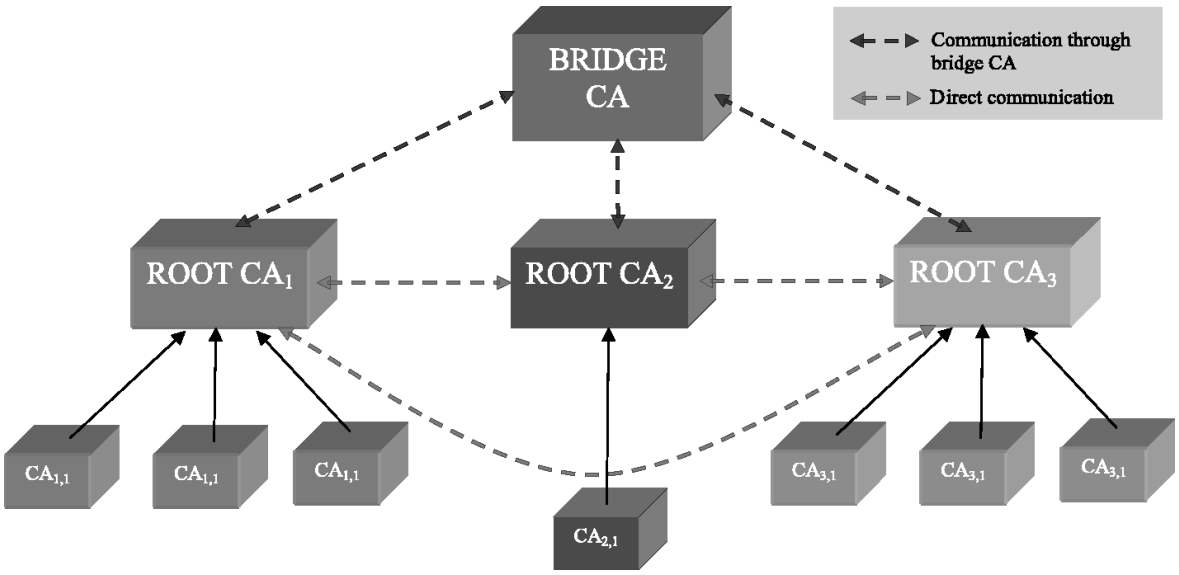
Le fonctionnement simplifié d'une ICP et les rôles de chaque partie sont décrits ci-dessous :

- **Structure :**

Une ICP est une structure hiérarchique de confiance.

La confiance est héritée à partir de la racine.

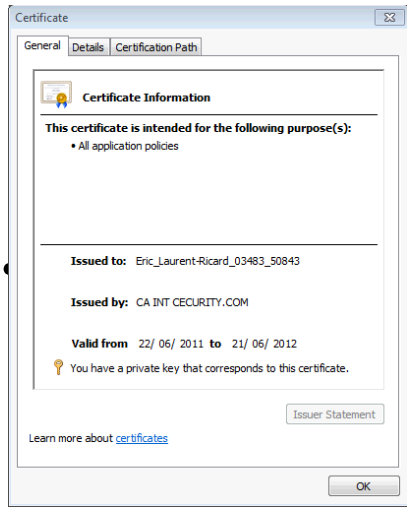
[12] ICP : Infrastructure à Clés Publiques (PKI en Anglais pour Public Key Infrastructure)

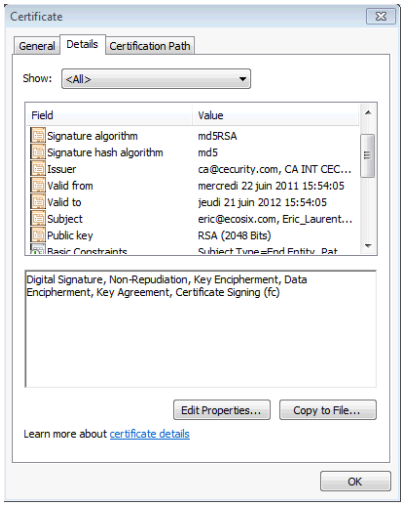


Source : Emerald Group Publishing Limited

Sous ce schéma se trouvent les utilisateurs des certificats émis.

Un certificat d'utilisateur est enregistré dans l'ordinateur et peut être consulté via les outils d'Internet Explorer par exemple :





La confiance est héritée de l'autorité supérieure, ainsi que le précédent schéma le montre, jusqu'à l'autorité racine (Root CA), à qui on choisit de faire confiance ou non.

AC : Autorité de Certification : Entité qui organise la confiance, édicte les règles de gestion et se porte garante des données émises.

AE : Autorité d'Enregistrement : Entité qui gère les données des utilisateurs (généralement en face à face) et s'occupe de la remise des certificats.

OC : Opérateur de Certification : Entité ayant en charge la gestion technique des certificats par délégation de l'AC.

- **Fonctions :**

L'AC rédige les règles de gestion des certificats, la délégation de confiance accordée aux AC déléguées, gère les contrats avec les AE et les OC.

C'est elle qui assume la responsabilité juridique issue de la confiance transmise aux acteurs.

L'AE est en charge de la remise des certificats, soit par logiciel, soit sur une clé (sécurisée ou non) à l'utilisateur du certificat.

Ces remises sont le plus souvent réalisées en « face à face » afin que l'AE puisse vérifier l'identité civile de l'utilisateur.

Elle assume la responsabilité de vérification de l'identité et la garantie de remise de la clé ainsi que la confidentialité et la traçabilité du processus de remise.

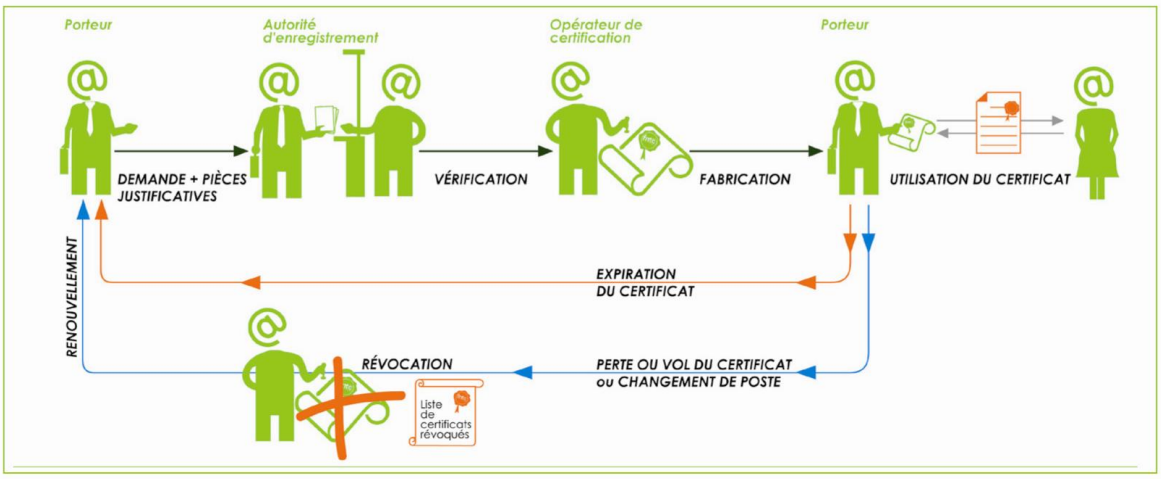
Elle gère parfois le support vis-à-vis de l'utilisateur final et les éventuelles demandes de révocation.

L'OC génère techniquement les certificats, les annuaires de diffusion des clés publiques, les listes de révocation et, le cas échéant, le « key escrow » qui permet de conserver des clés privés et de pouvoir les récupérer.

Elle a la responsabilité du fonctionnement technique, et en particulier de la haute disponibilité des serveurs et de la mise à jour des tables de révocation (CRL)[13].

Le schéma global du cycle de vie du certificat peut être représenté graphiquement de la façon suivante :

Le cycle de vie du certificat



Source : Guide la signature électronique de la FNTC (www.fntc.org)

[13] Certificate Revocation List

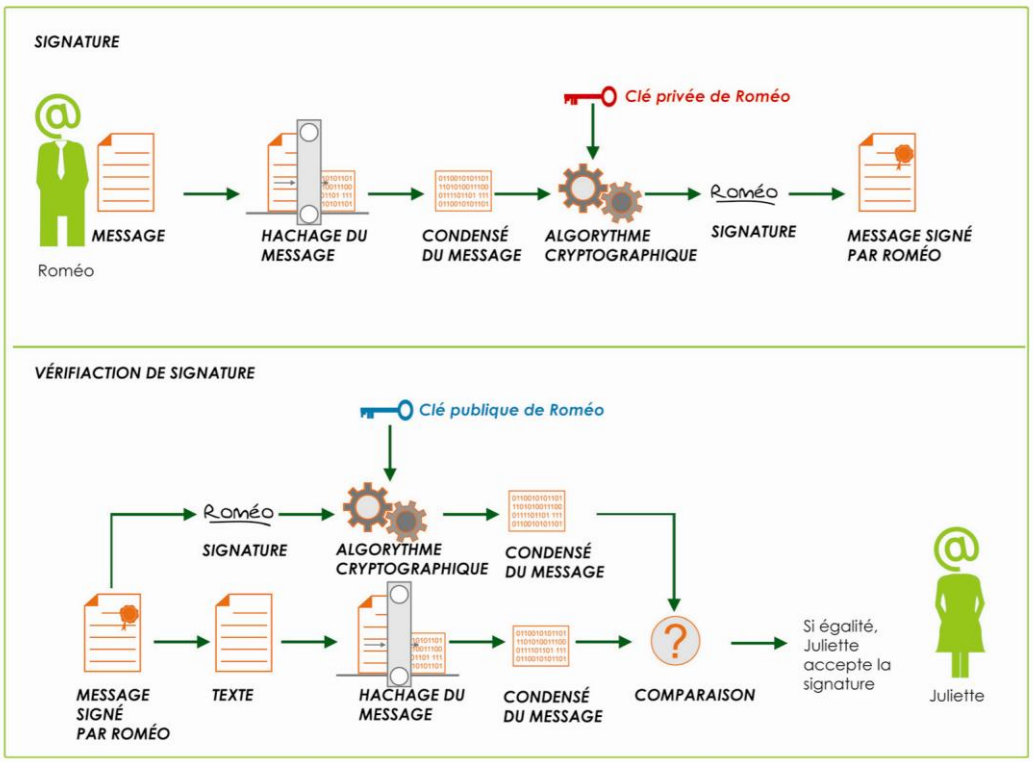
3. FONCTIONNEMENT DE LA SIGNATURE ELECTRONIQUE :

Le certificat de signature électronique peut avoir plusieurs buts et usages :

- La signature d'un document ou message,
- L'authentification du porteur,
- L'intégrité d'un document (qui est déjà inclus dans la signature),
- La confidentialité des documents ou messages,
- La non-répudiation d'une transaction.

L'objet de ce document n'est pas de détailler toutes ces fonctions dans leurs possibilités techniques et juridiques.

Pour simplifier la compréhension des mécanismes liés à la signature électronique proprement dite, le schéma suivant permet d'avoir une vue globale sur les différentes fonctions qui entrent en œuvre dans ce processus :



Source : Guide la signature électronique de la FNTC (www.fntc.org)

Les points importants à souligner ici sont :

- **Cryptographie asymétrique,**
- **Hachage (calcul d'empreinte) pour l'intégrité,**
- **Processus de vérification.**

VI. Points de faiblesses techniques des protocoles

1. LE PROTOCOLE SMTP

Parmi les points de faiblesse que l'on retrouve dans les caractéristiques du protocole SMTP, il faut noter principalement les suivants :

- **L'absence d'obligation de sécurisation du lien** : les connexions au serveur SMTP sont le plus fréquemment réalisées en clair, sans vérification de l'adresse du serveur, facilitant ainsi d'une part la possible récupération de mots de passe et d'autre part la possibilité de spoofing[14] ou d'attaque « man in the middle ».

- **L'authentification non systématique de l'émetteur sur le serveur** : On voit souvent des messages émis avec un nom apparent qui est un leurre, comme par exemple le nom du destinataire lui-même.

- **L'envoi de mots de passe en clair** : la connexion au serveur se fait sans sécurisation du lien, mais de plus, les mots de passe sont transmis tel quels directement, lorsque le serveur SMTP demande une authentification, ce qui est peu fréquent.

- **Un horodatage non fiable et facile à modifier** : Les dates et heures enregistrées dans les en-têtes des messages dépendent directement de l'heure des serveurs SMTP par lesquels ceux-ci transitent. Or, il est fréquent que ces serveurs ne disposent pas d'une heure correcte, ou celle-ci peut être aisément modifiée.

[14] Spoofing : se faire passer pour un ordinateur identifié

- **L'absence de fonction garantissant l'intégrité du message transmis** : Le protocole SMTP n'ajoute aucun mécanisme de vérification de la bonne réception des messages, et en particulier de leur intégrité qui peut être altérée au cours des différents relais.

- **La possibilité d'envoyer des messages sans nom ou adresse e-mail d'émetteur** : Pour des raisons de gestion des messages de retour d'erreur, le protocole SMTP impose la possibilité d'envoyer des messages sans nom d'émetteur, ce qui est une grave erreur et facilite l'anonymat du « spam ».

- **L'utilisation de la fonction VRFY pour obtenir des adresses e-mails distantes :** Cette fonction permet d'obtenir un retour du serveur SMTP de destination sur l'existence d'une adresse électronique sans même envoyer de message, ce qui autorise un robot à tester un ensemble important d'adresses dans un domaine jusqu'à ce qu'il obtienne des adresses valides, augmentant ainsi les possibilités de « spam ».

- **L'absence de vérification des DNS émetteurs** : Le protocole SMTP ne vérifie pas que l'émetteur et son éventuel serveur SMTP sont bien identifiés dans les tables DNS.

- **Les FQDN[15] qui ne sont pas toujours implémentés** : le chemin complet, normalement enregistré dans les DNS devrait être vérifié.

- **Pas de création systématique d'un identifiant de message unique** : Le protocole SMTP n'oblige pas à identifier de façon unique un message envoyé par un client de messagerie.

[15] FQDN : Fully Qualified Domain Name

Bien entendu, et ainsi que le déclare le RFC 5321 lui-même, la simplicité du protocole fait sa force et il n'est pas conçu pour gérer les aspects de sécurité ou d'identification.

Cela fait justement partie des points réfutés ici, et **la relative complexité des mécanismes proposés n'alourdiront pas, outre mesure, cette simplicité de fonctionnement, ni de développement du protocole.**

2. LE PROTOCOLE POP

De même, au niveau du protocole POP3, plusieurs options ou fonctions ne sont pas adaptées ou sont inexistantes pour assurer une meilleure confiance :

- L'horodatage n'est pas fiable et est facile à modifier, simplement en modifiant l'heure de son ordinateur,
- La fonction garantissant l'intégrité du message transmis n'existe pas,
- De même, il n'y a pas d'obligation de sécurisation du lien,
- La gestion des identifiants de messages n'est pas effectuée au niveau du protocole,
- L'envoi des login et mot de passe sont fréquemment effectués en clair, permettant une possible écoute et capture de ceux-ci.

« De futures extensions à POP3 sont en général déconseillées, car l'utilité de POP3 réside dans sa simplicité. POP3 est destiné à être un protocole de téléchargement et de suppression ; les capacités d'accès à la messagerie sont disponibles dans IMAP [IMAP4]. Les extensions qui prennent en charge l'ajout de boîtes aux lettres supplémentaires, permettent le téléchargement de

messages sur le serveur, ou qui dévient du modèle de téléchargement et suppression de POP sont fortement déconseillées et ont peu de chances d'être autorisées dans la perspective de la normalisation IETF. »
(Extrait de la RFC 1939)

Pour les mêmes raisons évoquées au chapitre précédent sur le protocole SMTP, **il est indispensable que les protocoles de base contiennent des fonctions systématiques assurant un minimum d'identification et d'intégrité des données.**

3. LE PROTOCOLE IMAP

Le protocole IMAP permet une gestion distante des messages qui restent conservés sur le serveur (MDA) sans charger le lien entre le serveur et le poste de travail du client.

Ce protocole avait son intérêt lorsque les liens Internet étaient gérés par des modems à basse vitesse, évitant des temps de transfert de messages important.

Par ailleurs, la conservation des messages par l'opérateur de messagerie permet d'alléger l'ordinateur local, et autorise une connexion à sa messagerie à partir de n'importe quel endroit.

IMAP contient de nombreuses possibilités de gestion des messages sur le serveur (le MDA), y compris au niveau des possibilités d'authentification (incluant Kerberos) et de sécurisation du lien.

D'autre part, ces sécurisations, tant du lien que de la confidentialité des mots de passe ne sont pas obligatoires et on peut retrouver des serveurs IMAP qui acceptent l'envoi de mots de passe en clair sur un lien non sécurisé.

Cela peut être un inconvénient, aussi bien vis-à-vis de la localisation du contenu de sa boîte de messagerie que vis-à-vis de la confidentialité des données.

En effet, l'évolution vers le « cloud computing » qui insiste sur la localisation répartie des applications et des contenus sur le web, ne permet pas de garantir un emplacement national de stockage, ni des moyens permettant de garantir la confidentialité des contenus ainsi répartis.

Il est donc important de conserver la possibilité de stocker localement, sur son poste de travail ou dans son réseau local le contenu de sa messagerie.

C'est ainsi que fonctionnent d'ailleurs les messageries d'entreprises comme Microsoft Exchange ou Lotus Domino, et dont le mécanisme apporte des garanties sur la gestion des boîtes aux lettres des utilisateurs.

Enfin, **les fonctions d'horodatage fiable et celles garantissant l'intégrité du message ne sont pas incluses dans ce protocole.**

4. LE « WEBMAIL »

Le fonctionnement du Webmail poursuit la même logique que le protocole IMAP en conservant systématiquement, et *a priori* sans limites de temps, l'ensemble des messages sur le serveur distant.

La facilité d'accès qu'apporte le Webmail de se connecter à partir de n'importe quel endroit, que ce soit avec son propre ordinateur ou une machine en libre service, ne doit pas faire oublier les contraintes et critères de sécurité, principalement au niveau de la sécurité du lien.

En effet, dès lors que l'on se connecte d'un site extérieur (cybercafé, accès wifi gratuit…), on prend encore plus de risque sur l'écoute des informations qui transitent sur le lien Internet.
Il est donc d'autant plus important que le lien d'accès au webmail soit sécurisé en SSL[16] avant d'envoyer son login et mot de passe.

Or ce fonctionnement n'est pas systématiquement proposé par les serveurs.

Par ailleurs, **les mêmes inconvénients que pour le protocole IMAP, concernant la localisation et la confidentialité des données existent à l'identique, de même que l'absence d'horodatage fiable et de l'intégrité des données.**

5. SUR TCP/IP

Le protocole de transport sur lequel s'appuient tous les protocoles de messagerie n'intègre pas de mécanisme de sécurité ni d'intégrité des paquets transportés.

De même, aucun horodatage fiable n'est assuré à ce niveau.

D'autre part, l'identification des émetteurs n'est pas assurée, à l'exception de leur adresse IP présentée qui peut être forgée, c'est-à-dire qu'un ordinateur peut utiliser une fausse adresse IP pour dissimuler l'origine des paquets émis, ce qui est fréquent dans le cas du « spoofing ».

On ne peut donc pas compter sur cette couche de transport pour assurer les fonctions d'identification d'authentification, de sécurité ou d'intégrité, et il faut donc adapter les protocoles de messagerie directement.

6. TRAÇABILITE

La traçabilité des messages électronique est contenue dans ses en-têtes et permet d'analyser par quels serveurs SMTP successifs un message est passé et à quelle date il est arrivé dans le MDA.

Cette traçabilité, intégrée dans le protocole SMTP est particulièrement simpliste et repose sur l'intégrité des serveurs SMTP utilisés ainsi que sur la qualité de la gestion de leurs horloges.

Un en-tête de message typique d'un « spam » est présenté ci-dessous :

```
Return-Path: <jonka@kissingerassoc.com>
Delivered-To: ericlau@business-models.com
Received: from b0.ovh.net (HELO queue) (213.186.33.50)
   by b0.ovh.net with SMTP; 30 Oct 2011 16:36:33 +0200
Received: from localhost (HELO mail519.ha.ovh.net) (127.0.0.1)
  by localhost with SMTP; 30 Oct 2011 16:36:33 +0200
Received: from b0.ovh.net (HELO queueout) (213.186.33.50)
   by b0.ovh.net with SMTP; 30 Oct 2011 16:36:33 +0200
Delivered-To: business-models.com-postmaster@business-models.com
Received: from b0.ovh.net (HELO queue) (213.186.33.50)
   by b0.ovh.net with SMTP; 30 Oct 2011 16:36:33 +0200
Received: from localhost (HELO mail519.ha.ovh.net) (127.0.0.1)
  by localhost with SMTP; 30 Oct 2011 16:36:33 +0200
Received: from b0.ovh.net (HELO queueout) (213.186.33.50)
   by b0.ovh.net with SMTP; 30 Oct 2011 16:36:33 +0200
Delivered-To: business-models.com-partenaires@business-models.com
Received: from b0.ovh.net (HELO queue) (213.186.33.50)
   by b0.ovh.net with SMTP; 30 Oct 2011 16:36:33 +0200
Received: from ps17321.dreamhost.com (69.163.205.29)
  by mx1.ovh.net with SMTP; 30 Oct 2011 16:36:28 +0200
Date: Sun, 30 Oct 2011 07:36:28 +0000
From: "Twitter" <notification-partenaires=business-
models.com@postmaster.twittercompletely.com>
Reply-To: noreply@postmaster.twittercompletely.com
To: partenaires@business-models.com
```

[16] SSL : Secure Socket Layer : protocole de sécurisation des liens Web

```
Message-Id:
<87824f732f73_75b085565ebba6c70@mx008.twittercompletely.com>
Subject: [SPAM] New notification from Twitter!
Mime-Version: 1.0
Content-Type: text/html; charset=utf-8
Content-Transfer-Encoding: Quoted-printable
Content-Disposition: inline
X-Campaignid: twitter20111030854317
Errors-To:      "Twitter"      <notification-partenaires=business-
models.com@postmaster.twittercompletely.com>
Bounces-To:      "Twitter"      <notification-partenaires=business-
models.com@postmaster.twittercompletely.com>
X-Ovh-Tracer-Id: 6983957123038832841
X-Ovh-Remote: 69.163.205.29 (ps17321.dreamhost.com)
X-Ovh-Local: 213.186.33.29 (mx1.ovh.net)
X-OVH-SPAMSTATE: SPAM
X-OVH-SPAMSCORE: 300
X-OVH-SPAMCAUSE:
gggruggvucftvghtrhhoucdtuddrfedvtddrvdduucetggdotefuucfrrhhofhhilhg
vmecuqfggjfenuceurghilhhouhhtmecufedttdenucfjughrucdvfeehleculdeftd
dtmd
X-Spam-Tag: YES (ovhvrmailscanner[300])
```

On distingue bien ici la première partie le « return path » qui est fréquemment indiqué avec une adresse inexistante dans le cas de « spam » de même que l'adresse, plus bas, « reply-to ».

Ensuite, le « received » permet de suivre le chemin emprunté par le message entre l'émetteur et le destinataire : ici, le message est passé par trois serveurs différents.

Il convient de remarquer la différence d'heure entre celle affichée dans le message 7h36 GMT et celle de connexion de l'émetteur au serveur 16h36 GMT+2.
Ceci montre que l'on ne peut pas compter sur ce système pour assurer une traçabilité correcte des messages transmis.

Ceci rend d'ailleurs la tâche de ceux qui doivent analyser la trace et l'origine des e-mails assez difficile.

Les autres champs des en-têtes informent sur le sujet du message, le type de contenu, puis d'autres champs ajoutés par certains serveurs SMTP qui analysent le contenu et/ou les émetteurs afin d'évaluer le niveau de « spam » du message.

Il convient donc d'apporter des modifications substantielles à certains RFC pour obtenir une meilleure traçabilité des messages, qui est également liée à l'identification de l'émetteur.

VII. Une nouvelle architecture de confiance

Afin de pouvoir établir une nouvelle architecture de confiance, il faut établir la liste des fonctions et services nécessaires.

La confiance est une chaîne qui doit être ininterrompue de l'émetteur au destinataire, et même au-delà dans la durée de conservation.

1. COMMENT IDENTIFIER UNE SOURCE

La source d'un message correspond à l'ordinateur à partir duquel le message a été émis.

L'identification de celui-ci est réalisée par **la connaissance de son adresse IP** qui, normalement, informe également sur sa localisation.

Cette identification de la source d'un message est difficile à obtenir dans l'environnement actuel de l'utilisation des protocoles de l'Internet.

En effet, les adresses IP des émetteurs de messages ne sont pas toujours « parlantes », car souvent noyées au sein d'un bloc d'adresses privé ne permettant pas l'identification d'un ordinateur précis, mais seulement le routeur d'entrée de l'entreprise. Ceci nous donne donc une information limitée, mais généralement réelle.

Le cas d'utilisation d'un accès ouvert proposé par un hôtel ou restaurant par exemple, ne permet pas d'identifier la source d'un message, mais seulement le lieu à partir duquel le message a été envoyé, car l'ordinateur utilisé obtient une adresse IP spécifique dépendant du lieu de connexion.

Plus difficile encore, le « spoofing » d'adresse IP est courant chez les spammeurs et dans les réseaux de zombies (botnets) utilisés pour diffuser massivement le « spam ». Dans ce cas, la donnée d'adresse IP émetteur n'a plus aucun sens et on ne peut ni identifier, ni localiser la source du message.

Prenant en considération l'ensemble de ces problèmes, le MAAWG (Messaging Anti Abuse Working Group), qui travaille sur les moyens de réduire le "spam" écrit :

> *"**Trust in Email Begins with Authentication (June 2008)**"*
>
> *"« Les mécanismes d'authentification peuvent aider à distinguer le courrier électronique légitime du pourriel. Lorsqu'ils sont utilisés comme partie d'un programme anti-abus à multiples facettes, ils deviennent un outil efficace pour aider à protéger les marques commerciales de la contrefaçon et les attaques de hameçonnage », a déclaré Dave Crocker, conseiller principal de MAAWG.*
>
> *Les mécanismes d'authentification de courrier électronique sont utilisés pour valider l'identité d'un expéditeur de message, en étouffant ainsi les prétendus polluposteurs qui faussent souvent le champ 'De' du courrier électronique pour déjouer les mesures de détection. "*

Ainsi que nous l'avons vu au *chapitre II*, un protocole comme **DKIM**[17] **permet d'identifier de façon assez sécurisée le serveur de messagerie de l'émetteur** et de garantir que ce serveur est bien celui qui est référencé dans le DNS du domaine correspondant.

En conséquence, si un utilisateur essaye d'utiliser le serveur SMTP d'un autre domaine que le sien, celui-ci sera détecté et le destinataire en sera informé.

Par ailleurs, l'identification de l'adresse IP de l'émetteur du message est transmise au serveur SMTP qui devrait vérifier si sa valeur est bien en relation avec le domaine auquel il appartient afin d'éviter le « spoofing » d'adresse IP.

Cette vérification a ses limites, car l'adresse IP d'un utilisateur itinérant pourra varier fréquemment alors même que son message sera « valide ».

2. AUTHENTIFIER L'EMETTEUR

L'émetteur d'un message électronique est identifié par son adresse e-mail, et non par son identité civile, même si, dans le cas de certificats de signature électronique de type II ou III son identité civile est incluse et validée par une autorité de confiance.

Au-delà de l'identification de la source, **il convient donc d'authentifier l'émetteur du message**, ce qui a beaucoup plus de valeur que l'identification de la source.

Il est nécessaire de préciser, ici, la distinction faite entre **l'identification** d'un ordinateur par exemple, avec **l'authentification** de l'utilisateur qui se trouve derrière l'ordinateur et qui envoie le message.

En effet, si, par l'adresse IP, même valide et vérifiée, on peut **identifier** l'ordinateur ayant émis un message, on ne peut garantir qui a utilisé cet ordinateur. Ceci est particulièrement vrai dans le cas d'utilisation des cybercafés ou de points d'accès publics.

[17] Domain Keys Identified Mail

Par contre, si on peut **authentifier** l'utilisateur qui émet le message, quelque soit l'ordinateur qu'il utilisera pour envoyer son message, nous disposerons d'une information fiable.

Cette authentification est d'autant plus importante qu'il est fait référence, dans les dispositions du Code civil à une identification précise de la personne dont émane un document ou un message.

Autrement dit, **la meilleure façon d'authentifier l'émetteur d'un message est de banaliser et généraliser l'utilisation de la signature électronique simple** (Classe I).

Ce niveau de certificat ne permet que de **valider l'adresse e-mail du demandeur** et non pas son identité civile.
Cette création de certificat et la gestion des clés correspondantes peut être simplifiée dans l'idée de ce simple objectif.

Or, sans chercher à garantir que le logiciel de messagerie de « Monsieur Martin » n'a pas été utilisé à son insu, on peut, pour le moins, garantir que la signature électronique qu'il a utilisée est bien celle de son adresse e-mail.

Il peut s'agir aussi bien de certificat électroniques de type X509 comme ceux couramment utilisés chez les opérateurs de confiance français ou étrangers, que des certificats de type PGP dont la confiance n'est pas issue d'une structure de type PKI, mais par une confiance de proximité, de connaissance en connaissance.

Les deux solutions techniques ont leurs avantages, et la solution de confiance à mettre en place doit accepter les deux types de certificats.

L'aspect important dans ce cas est de garantir que l'adresse e-mail utilisée par l'émetteur d'un message existe, et que c'est bien à partir de celle-ci que le message a été envoyé.

3. SOLUTIONS DE SECURISATION

Les échanges entre l'émetteur et le serveur de messagerie, entre les serveurs de messagerie successifs, puis, enfin, avec le client de messagerie du destinataire doivent être sécurisés et cryptés afin de ne pouvoir intercepter le contenu des échanges (soit par écoute ou par attaque de type « man in the middle »), et de pouvoir garantir l'authenticité des serveurs utilisés.

La sécurisation du lien entre le poste de travail et le MDA (Serveur POP ou IMAP ou Webmail) **doit être systématiquement mise en place**.

Au-delà de l'établissement du lien TLS, il faut ajouter une vérification automatisée du certificat utilisé par le serveur afin de l'identifier de façon quasi certaine.

L'accès à un serveur LDAP fournissant un annuaire garanti par une Autorité de Certification, permet à l'émetteur de vérifier que son serveur de messagerie (MTA) est bien le sien, et donc d'éviter un éventuel « spoofing » de celui-ci.

Du côté du serveur (MTA), il lui faut également vérifier, auprès du même serveur LDAP l'existence et la validité de l'e-mail de l'émetteur et de son certificat.

Une fois la sécurité du lien établie, il convient de fiabiliser l'authentification de l'utilisateur qui se connecte pour accéder à sa boîte de messagerie en utilisant les extensions adaptées des protocoles (POP ou IMAP par exemple).

Les extensions APOP cryptant simplement le mot de passe lors de son envoi, ou POP-AUTH qui utilise une méthode de « challenge-response » pour s'assurer de la validité du mot de passe sont trop limitées.

L'utilisation du certificat de signature électronique de classe I permettra d'authentifier l'utilisateur lors de sa connexion au serveur POP et remplacera le mécanisme habituel de « login-password », lui garantissant ainsi qu'il sera le seul à pouvoir y accéder.

Dans le cas d'un utilisateur itinérant, la copie de son certificat et de ses clés publiques et privées sur un support externe sera possible dès lors qu'il prendra les précautions nécessaires à la conservation sécurisée de ceux-ci.

Enfin, il convient de sécuriser et de rendre confidentiel le stockage des messages sur les MDAs afin que, même les administrateurs de ces systèmes ne puissent avoir accès à ces données (c.f. §7).

4. GARANTIR L'INTEGRITE DU MESSAGE

Un message « intègre » est un message dont le contenu n'a pas été altéré.

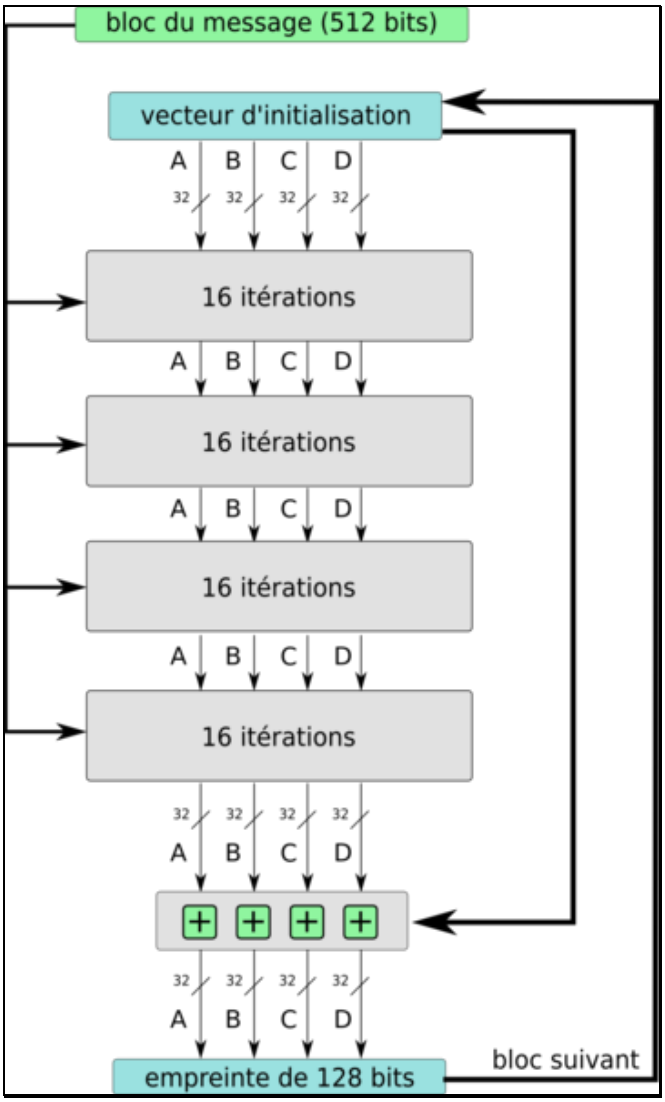
Compte tenu du fait qu'il est impossible de reconstituer un message d'origine qui aurait été altéré, on doit se contenter de la garantie de la vérification de l'intégrité du contenu.

Celle-ci est réalisée grâce à un mécanisme de calcul d'empreinte (hash) appelé aussi condensé ou empreinte cryptographique permettant d'identifier de façon quasi-unique un document quelconque à l'aide d'une valeur de 160 bits pour la fonction SHA-1 par exemple.

Si un seul bit ou un caractère a été changé dans le document, son empreinte sera sensiblement modifiée, et nous obtiendrons la preuve de la modification de celui-ci.

Ce principe de calcul d'empreinte (hash) est essentiel du point de vue de l'intégrité des document, et est utilisé dans de nombreux domaines comme la signature électronique, l'horodatage, la sécurisation d'échanges...

A titre d'exemple, le schéma de calcul de SHA-1 est le suivant :



Cette intégrité du message doit être garantie de bout en bout de la chaîne de transmission.

Pour ce faire, même si l'utilisation de la signature électronique simple côté émetteur est la meilleure solution, celle-ci n'est pas suffisante, car elle ne concerne que le contenu lui-même du message.

Or, nous avons vu que des paramètres sont ajoutés à chaque étape de la transmission du message par les différents serveurs SMTP qui se chargent du transfert du message.

En effet, les en-têtes des messages sont modifiés à chaque passage par un « relais » de messagerie, et ce jusqu'au destinataire final.
Chacun de ces serveurs ajoute, au moins, les informations d'horodatage et d'adresse IP du serveur.
Souvent d'autres informations sont ajoutées telles qu'une identification de type DKIM, une analyse du niveau de « spam »…

En conséquence, **à chaque étape une vérification d'intégrité doit être réalisée**, puis le serveur ajoute ses paramètres, et **génère un nouvel élément permettant de garantir l'intégrité des données qu'il a ajouté**.

Cela peut se faire soit en recalculant une empreinte (hash) globale du message avec ses en-têtes, soit en calculant plus simplement une empreinte complémentaire liée uniquement aux données ajoutées par le serveur.

Dans tous les cas, l'utilisation du calcul d'empreinte est la seule manière fiable et standardisée d'assurer l'intégrité des données.

A ce titre, et compte tenu des évolutions de la technologie, il faut utiliser, *a minima*, le mode SHA-1 ou SHA-256 pour se garantir, tant que possible, d'éventuelles « collisions » d'empreintes (deux fichiers différents donnant la même empreinte).

Idéalement, l'empreinte est calculée à partir du certificat électronique du serveur SMTP permettant ainsi de garantir l'identité de celui-ci.

5. SOLUTIONS DE TRAÇABILITE

Pour garantir l'origine d'un message, il faut être capable de retracer son chemin, et pour cela, le contenu des en-têtes des messages sont essentiels.

La solution de traçabilité des messages s'appuie tout d'abord sur l'intégrité de ceux-ci, en utilisant les méthodes décrites dans le paragraphe précédent.

Mais, au-delà de cette technique, il convient également de **garantir un horodatage fiable** des messages qui transitent.

En effet, il est facile de modifier la date et l'heure d'un serveur dont on a le contrôle afin d'enregistrer des données d'horodatage erronées (volontairement ou non) dans les en-têtes des messages.

Cela implique d'une part **l'utilisation de serveurs d'horodatage sécurisés** (TSS : Time Stamping Server) intégrés ou non au serveur de messagerie SMTP, ainsi que la modification du format des dates enregistrées dans les en-têtes des messages pour se conformer aux standards des formats d'horodatage que sont d'une part le RFC3161 et la norme ISO 8601, et d'autre part de pouvoir disposer d'une identification quasi certaine du serveur de transit ou d'envoi.

A ce titre, **l'utilisation de certificats électroniques sur les serveurs de type MTA faciliterait à la fois la mise en œuvre de l'intégrité et de la traçabilité des messages.**

Par ailleurs, il conviendrait pour ces mêmes serveurs, de mettre en place une vérification automatisée des adresses IP présentées par un « reverse DNS » car l'absence d'un nom d'ordinateur associé à une adresse IP est un premier signe laissant penser à une utilisation détournée de la messagerie.

Dans le cas où chaque serveur MTA utiliserait des certificats électroniques, la recherche dans l'annuaire LDAP de l'Autorité de Certification correspondant permettrait facilement d'assurer la validité du serveur.

6. ENVISAGER L'ARCHIVAGE SUR LE LONG TERME

La valeur des informations de transit des messages pour les serveurs d'une part, et celle des messages électroniques eux-mêmes d'autre part, induit un besoin réel de sauvegarde sécurisée de ces éléments.

Plus encore, un archivage sur le long terme, lui-même sécurisé, de ces messages permettra de disposer de documents qui auront une valeur probante au regard des dispositions du Code civil.

Cet archivage sera lui-même garanti par des certificats électroniques, horodatages et calculs d'empreintes.

L'archivage des données de transit (logs) des serveurs sera également indispensable sur une durée qui sera fonction des contraintes de conservation juridique de ces traces qui sont spécifiques à chaque pays.

Ces conditions de conservation doivent également tenir compte des spécifications relatives aux données personnelles telles que le « droit à l'oubli » défini par la CNIL en France.

Le cas échéant, il sera ainsi possible de retracer le « parcours probant » d'un message et de le vérifier en comparant ces traces au contenu des en-têtes du message reçu par l'utilisateur.

7. SOLUTIONS DE CONFIDENTIALITE

La confidentialité réside dans l'impossibilité pour un tiers de lire le contenu d'un message ou d'un document.

Un premier niveau doit être assuré lors du transit des messages, ainsi que nous l'avons vu au §3.

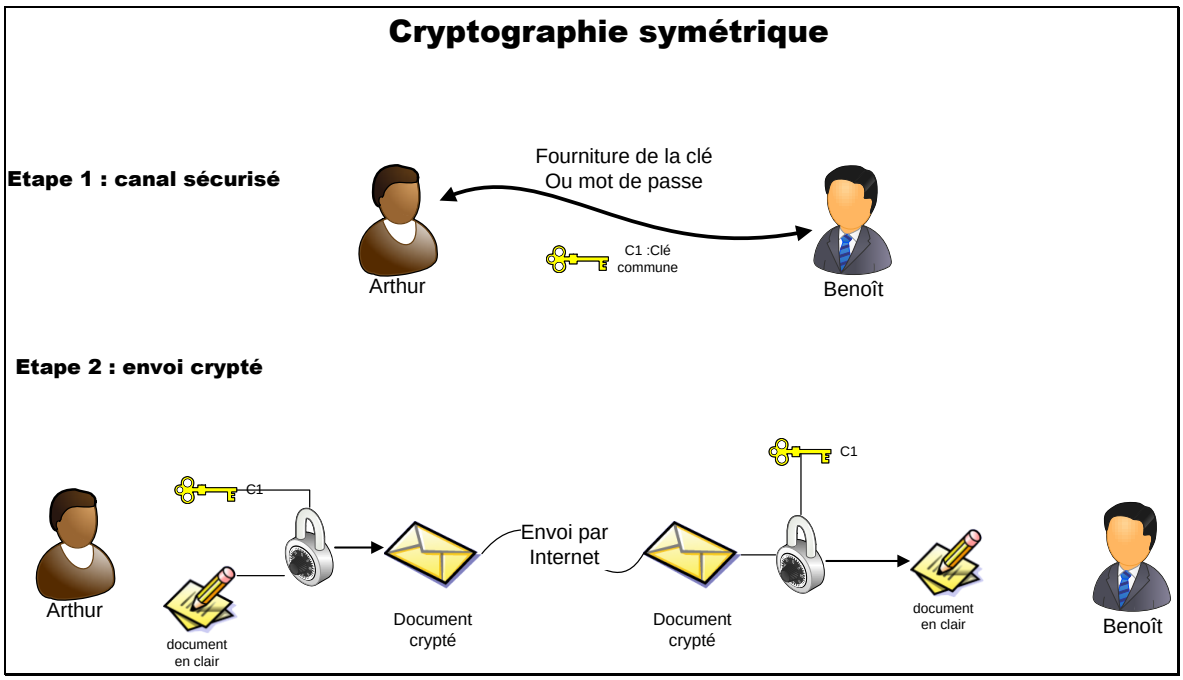
Des solutions au niveau global d'un serveur de messagerie de type MDA sont nécessaires afin qu'un tiers pouvant avoir accès à ce serveur ne puisse lire les données de l'utilisateur.

Par ailleurs, vis-à-vis de messages spécifiques, un deuxième niveau de confidentialité peut être appliqué localement par l'utilisateur, soit à partir de son certificat de signature électronique, soit à partir d'une clé de codage symétrique.

Ces différences de fonctionnement nécessitent une courte explication sur les fonctionnements des deux modes courant de cryptage que sont le cryptage symétrique et le cryptage asymétrique.

- **Fonctionnement de la cryptographie symétrique :**

Le schéma suivant permet de présenter de façon simple le mécanisme de la cryptographie symétrique :



Dans une première étape, l'émetteur doit fournir au destinataire, de façon sécurisée, la clé qui sera utilisée pour crypter les données (document ou message). Ceci peut être fait par remise en mains propres sur un support physique, de vive voix, par courrier…

Ensuite, lorsque l'un ou l'autre veut envoyer un message crypté, il utilise leur clé commune ainsi qu'un logiciel identique, ou du moins cryptant les données de la même manière, pour créer un document ou un message confidentiel.

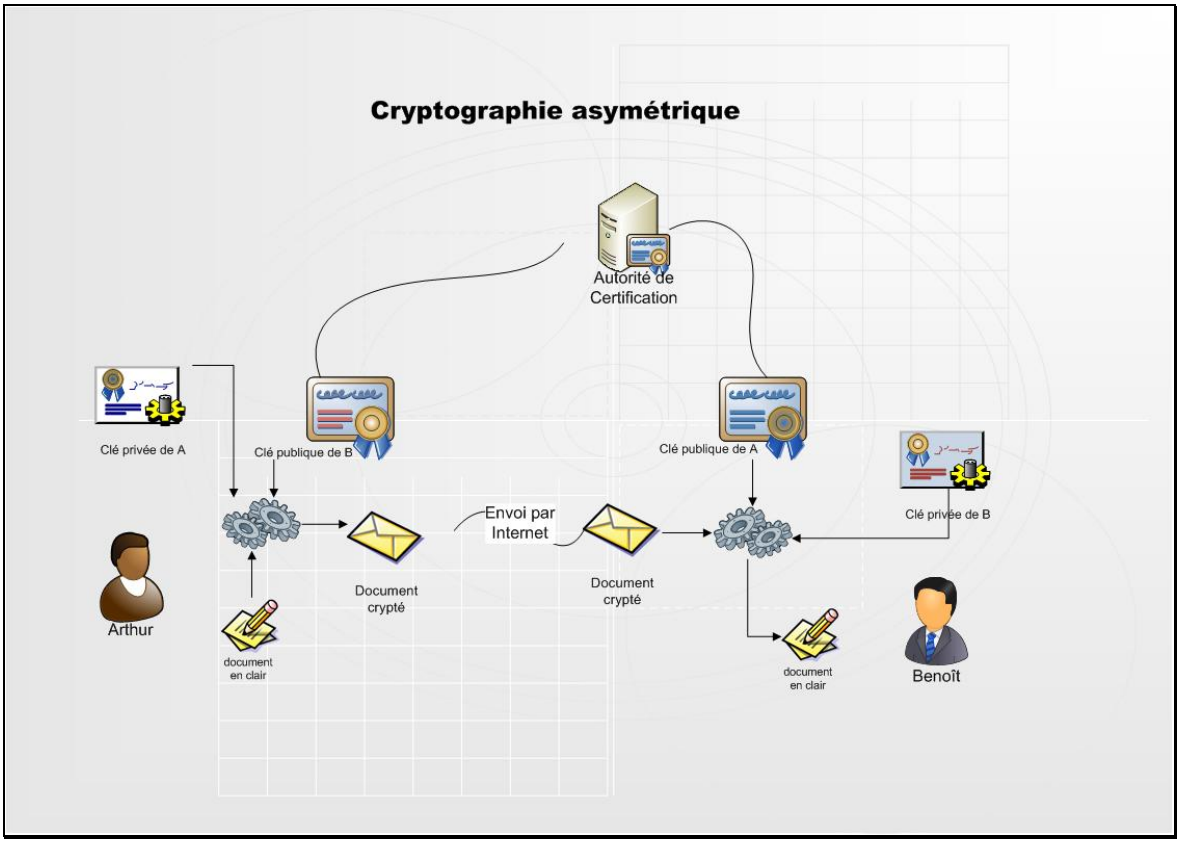
Celui-ci peut être ensuite envoyé par n'importe quel canal non sécurisé, et sera ensuite décrypté par le destinataire avec la même clé.

Bien entendu, ce mécanisme a ses limites, que ce soit un risque d'oubli du mot de passe, ou au contraire l'enregistrement de celui-ci sur un ordinateur auquel un tiers pourra accéder et obtenir cette clé de cryptage.

Ses principaux avantages sont la simplicité du fonctionnement et la performance pour rendre des données confidentielles.

- **Fonctionnement de la cryptographie asymétrique :**

Le mécanisme de la cryptographie asymétrique suppose que l'émetteur et le destinataire disposent déjà de certificat électronique type X509[18] par exemple.



L'émetteur A souhaite envoyer un message crypté à destination de B.

[18] X509 : Norme internationale des certificats de signature électronique

A et B disposent de certificats électroniques dont la clé privée est conservée par chacun d'entre eux, et la clé publique est disponible sur un serveur de l'autorité de certification, ou est transmise à l'autre partie directement.

Pour pouvoir utiliser ce mécanisme, il faut, en outre, que le certificat électronique délivré par l'autorité de certification autorise la fonction de cryptage d'un document ou d'un message.

Or, il convient de remarquer, ainsi que je l'ai expliqué au chapitre IV.5, que très peu d'autorité de certification accepte de valider cette fonction dans leurs certificats.

Ceci veut dire que cette fonction doit être développée au niveau des tiers de confiance pour permettre d'assurer une meilleure confidentialité des données.

Notre émetteur A disposant donc d'un certificat de signature électronique ET de cryptage des données, peut alors utiliser les fonctions intégrées dans son logiciel de messagerie par exemple pour crypter son message.

Le principe essentiel de ce cryptage des données réside dans le fait que la clé publique du destinataire est utilisée, conjointement à la clé privée de l'émetteur pour crypter le message.

En conséquence, SEUL le destinataire pourra décrypter le message avec sa clé privée.
Il utilisera la clé publique de l'émetteur pour vérifier la provenance et l'authentification de celui-ci.

Les inconvénients liés à ce mécanisme incluent la mise à disposition de certificats permettant de crypter des données, la relative lenteur du cryptage et la disponibilité du serveur de l'AC pour les vérifications.

A l'image du protocole SSL utilisé par la plupart des serveurs Web pour sécuriser un lien, il est possible d'utiliser la cryptographie asymétrique pour échanger, de façon sécurisée, un mot de passe ou une clé de cryptographie symétrique qui pourra être utilisée en toute confiance.

Dans le cadre de cette nouvelle architecture de confiance, il convient de disposer, justement, de certificats de signature électronique autorisant également le cryptage.

Une fois ceux-ci déployés, il devient possible d'assurer la confidentialité des données, soit en utilisant simplement la cryptographie asymétrique, soit en combinant cette dernière avec la cryptographie symétrique.

- **Solution de durée de vie d'un message :**

Dans le cadre de la confidentialité, ou même dans un cadre marketing, il est intéressant de donner une durée de vie limitée à un message.

Pour ce faire, on utilisera plutôt un cryptage symétrique en faisant héberger la clé de décryptage sur un site distant tel qu'un annuaire LDAP[19] pendant une durée limitée.

Ceci aura pour effet de ne pouvoir décrypter ledit message que pendant une période précise. Ensuite, la clé sera effacée et le message ne pourra plus être décodé.

Bien entendu, cela n'empêche pas l'utilisateur ayant décodé le message pendant sa durée de vie, de le copier, décrypté, puis de le stocker sous cette dernière forme. Néanmoins, dans ce cas, les attributs de traçabilité et d'intégrité pourraient ne pas être conservés.

[19] LDAP : Lightweight Directory Access Protocol

- **Sécurisation des données stockées :**

Les messages électroniques sont stockés sur les MDAs dans l'attente de la connexion de l'utilisateur avec son client de messagerie.

De plus en plus fréquemment, les utilisateurs conservent leurs messages sur ces serveurs, en particulier dans le cas de l'utilisation du « Webmail ».

Or, dans la plupart des cas, n'importe quel ingénieur système disposant d'un accès privilégié aux ordinateurs hébergeant les MDAs, peut obtenir le contenu d'un message stocké sur celui-ci.

Ce point est d'autant plus critique, que ces serveurs sont hébergés dans des pays dans lesquels la protection des données personnelles ne s'applique pas.

Aussi, il convient de sécuriser les messages électroniques stockés sur ces serveurs afin que personne ne puisse accéder à leur contenu en-dehors de l'utilisateur lui-même.

Ceci implique, *a priori*, un cryptage des données sur ces serveurs.

Afin d'en assurer la confidentialité pour l'utilisateur final, la solution préconisée sera d'utiliser la clé publique de l'utilisateur sous réserve de son existence.

Dans ce cas, l'utilisateur utilisera sa clé privée pour accéder à ses données déportées et confidentielles.

8. CONSEQUENCES SUR LES PROTOCOLES

Historiquement des tentatives d'amélioration du logiciel « sendmail », le plus couramment utilisé sur Internet comme serveur SMTP, ont eu lieu avec le développement de la version « sendmail X » qui devait prendre en compte plusieurs aspects de sécurisation et d'identification des émetteurs.

Ce projet a été arrêté en 2007 et remplacé par le projet « MeTA1 »[20] mais en se focalisant sur la fiabilité et l'efficacité du protocole, plutôt que sur les aspects liés à l'authentification des émetteurs.

Or, pour assurer la mise en œuvre des recommandations précédentes sur **l'identification de la source, l'authentification de l'émetteur, la sécurisation des transferts, l'intégrité et la traçabilité des messages ainsi que la confidentialité**, il convient tout d'abord **d'adapter le protocole SMTP** pour lui permettre d'assurer, au niveau du protocole lui-même, ces différentes fonctions.

Une nouvelle version du protocole, que j'appellerai pour l'instant CEMTP pour CErtified Mail Transfer Protocol, pourrait ainsi devenir le « standard de fait » des MTAs afin d'élever sensiblement le niveau de confiance dans la messagerie.

De même, il convient d'adapter les contraintes liées aux MDAs (serveurs POP / IMAP) afin de **garantir ces éléments d'intégrité, d'authentification, de traçabilité et de confidentialité de bout en bout et d'imposer l'utilisation de certaines extensions déjà existantes.**

[20] http://www.MeTA1.org/

Plus précisément, certaines fonctionnalités du protocole POP3 devraient être obligatoires et non optionnelles telles que l'authentification avec un certificat, ou pour le moins une séquence « challenge-response » de type POP-AUTH, et **d'autres devraient être ajoutées comme la sécurisation du lien avec le serveur SMTP et surtout la confidentialité des données sur le serveur.**

Enfin, pour gérer, du point de vue de l'utilisateur, l'ensemble de ces contraintes et de ces protocoles, tout en gardant une rétro-compatibilité, il est indispensable de **développer un « client » de messagerie (MUA) permettant une gestion simple et transparente de tous ces critères : *« certitrustmail »* par exemple.**

Celui-ci aura pour objectifs :

- l'accès simplifié à la demande et à l'utilisation de certificats de signature électronique de Classe I (e-mail**) permettant le cryptage des données,**
- L'authentification de l'utilisateur pour ses connexions à l'aide de ce certificat,
- la gestion de la confidentialité des données localement et à distance,
- L'établissement d'un lien sécurisé avec le serveur CEMTP,
- La connexion automatisée aux serveurs LDAP gérant les clés publiques et la vérification systématique des certificats,
- La séparation des fonctions de gestion des messages électroniques,
- La mise en œuvre des mécanismes d'intégrité des messages, même en dehors de l'utilisation des serveurs CEMTP,
- La gestion d'un horodatage fiable permettant d'assurer une bonne traçabilité.

VIII. Propositions de modifications des protocoles

1. QU'EST-CE QU'UN RFC ?

Les RFC (Request For Comment) représentent la manière de normaliser les protocoles de l'Internet.

Le fonctionnement des RFC est le suivant :

« Les RFC sont rédigées sur l'initiative d'experts techniques, puis sont revues par la communauté Internet dans son ensemble. Cela diffère d'une publication d'institution telle que l'ANSI.

La majorité des RFC utilisent les termes MUST, MUST NOT, SHOULD, MAY, etc. tels que définis dans la RFC 2119[4] pour définir leurs exigences (obligation, interdiction, recommandation, etc.). Pour plus d'informations à propos des RFC et les procédures associées, voyez la RFC 2026[5] « Procédures Standards d'Internet. Révision 3 ».

Les RFC font d'abord l'objet d'un draft (brouillon). Tout le monde peut écrire un draft. Ils n'ont donc aucune valeur. Après avoir écrit un draft, on peut le soumettre à l'IETF en le transmettant à rfc.editor@rfc.editor.org. Tous les drafts n'étant pas dignes d'intérêt, ils ont une date de péremption. Si le draft attire l'intérêt de la communauté, un groupe de travail peut être créé pour la rédaction d'une RFC. La RFC 2223[6] donne les instructions pour les futurs auteurs.

Quelques RFC finissent par devenir des standards d'Internet. La procédure complète pour la transcription d'une RFC en standard est la suivante :

RFC → Proposed Standard → Draft Standard → Internet Standard

Malgré leur nom, les RFC sont le plus souvent stables. Toute modification apportée à une RFC entraîne l'écriture d'une nouvelle RFC, qui rend la précédente obsolète. »

Source : Wikipedia : simplification traduite des informations du site de l'IETF.

2. QUELLES NORMES EXISTENT ?

Les RFC, c'est-à-dire l'IETF[21] bien entendu, mais aussi l'ISO (X509 ; PDF/A…), l'ETSI, … sont les organismes qui rédigent des normes sur le fonctionnement de l'Internet.

Néanmoins, compte tenu du nombre important de ces standards et normes, nous nous limiterons aux plus significatifs qui concernent le sujet même de cette thèse, en particulier les protocoles de messagerie principaux qui sont les suivants :

- SMTP : RFC 5321 (anciennement 2821)
- POP3 : RFC 1939
- IMAP4 : RFC 3401
- Structure des messages : RFC 2822
- En-têtes des messages : RFC 5322

3. DEMANDER LA MODIFICATION D'UN RFC

Une demande de publication d'un RFC commence par la rédaction de celui-ci selon les formats définis dans la **RFC 2223** qui la structure.

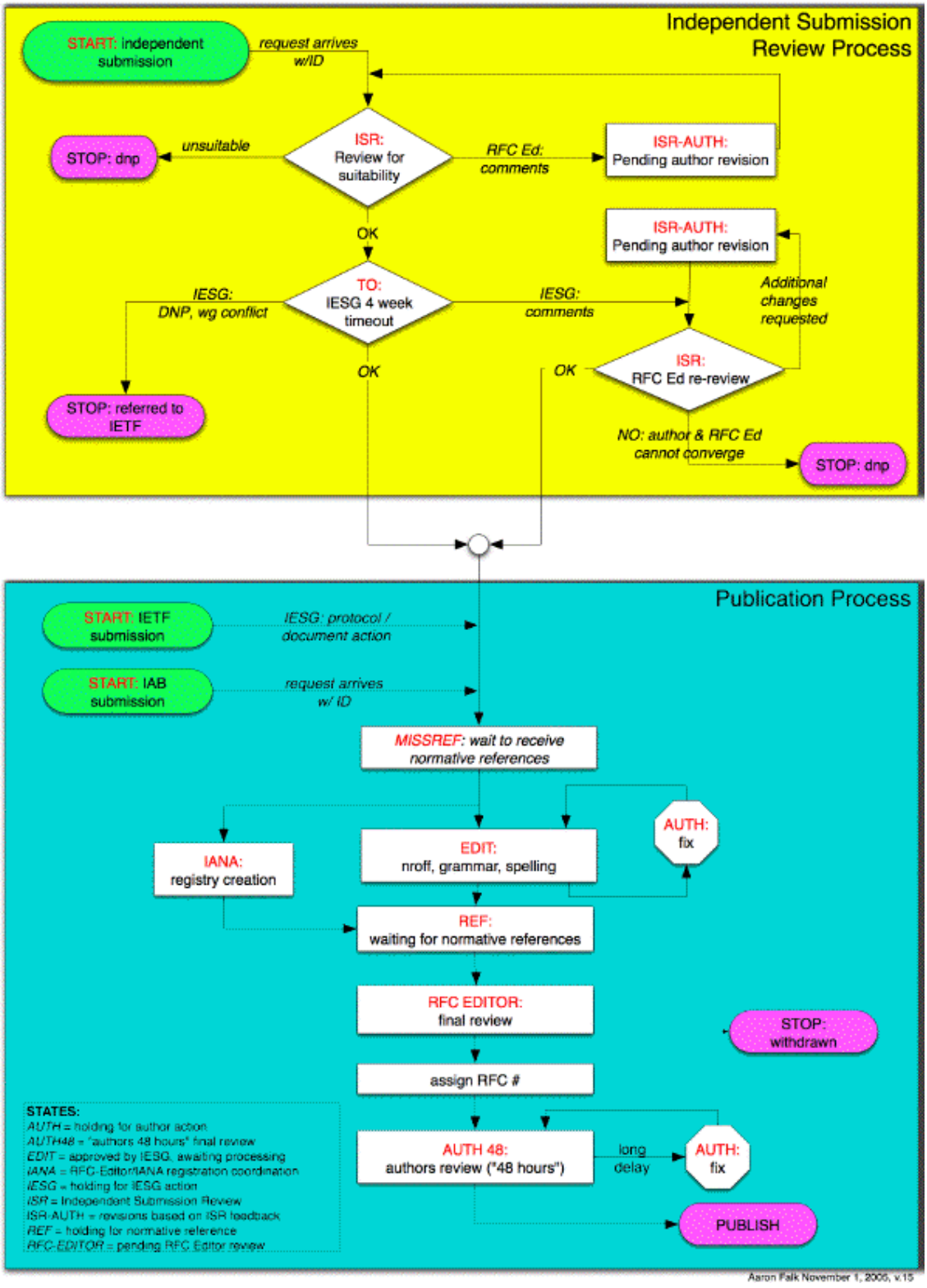
Il convient alors d'envoyer sur le site de l'IETF, une première version qui sera un « draft » et sera revue par des membres de l'IETF, comportant, le cas échéant plusieurs échanges pendant 6 mois.

Ensuite, celle-ci peut être proposé dans la **« file d'attente d'édition »** dont le schéma est le suivant :

[21] IETF : Internet Engineering Task Force : www.ietf.org

RFC Editor

Document Processing



A l'issue de ce processus, et s'il est positif, un numéro de « RFC » est attribué.

Le document est alors publié à l'ensemble des membres des listes de diffusion correspondant au domaine d'application, dans notre cas, la messagerie électronique, et qui comportent la plupart des spécialistes mondiaux du domaine.

Le document devra être revu à la lumière des retours et critiques de ces membres.

Enfin, si les retours sont positifs et si les organes de gestion de l'IETF considère que le document le justifie, il devient un RFC définitif.

4. PROPOSITIONS POUR LE PROTOCOLE CEMTP

Ainsi que nous l'avons vu dans le chapitre VII, un certain nombre de points et de fonctionnalités doivent être ajoutées et/ou modifiées dans le protocole SMTP pour le transformer en protocole original CEMTP.

Pour ce faire, les critères à retenir sont les suivants :

- Sécurisation des échanges
- Identification de la source
- Authentification de l'émetteur
- Intégrité des messages de bout en bout
- Traçabilité
- Confidentialité des données
- Archivage des logs (traces) de façon sécurisée

En premier lieu, **l'établissement de liens sécurisés lors des échanges**, que ce soit entre l'utilisateur et le serveur (MUA et MTA), ou entre deux serveurs (MTAs), **DOIT être systématique.**

Ensuite, il convient d'assurer **l'identification du serveur de messagerie** de l'émetteur, pour les relais ou les réceptions de messages.
A ce titre le protocole existant **DKIM est parfaitement adapté et devrait être utilisé.**

Détail du RFC 4871 expliquant DKIM :

> DomainKeys Identified Mail (DKIM) defines a mechanism by which email **messages can be cryptographically signed, permitting a signing domain to claim responsibility for the introduction of a message into the mail stream.** Message recipients can verify the signature by querying the signer's domain directly to retrieve the appropriate public key, and thereby confirm that the message was attested to by a party in possession of the private key for the signing domain.
>
> The approach taken by DKIM differs from previous approaches to message signing (e.g., Secure/Multipurpose Internet Mail Extensions (S/MIME) [RFC1847], OpenPGP [RFC2440]) in that:
>
> o the message signature is written as a message header field so that neither human recipients nor existing MUA (Mail User Agent) software is confused by signature-related content appearing in the message body;
>
> o there is no dependency on public and private key pairs being issued by well-known, trusted certificate authorities;
>
> o there is no dependency on the deployment of any new Internet protocols or services for public key distribution or revocation;
>
> o signature verification failure does not force rejection of the message;
>
> o no attempt is made to include encryption as part of the mechanism;
>
> o message archiving is not a design goal.
>
> DKIM:
>
> o is compatible with the existing email infrastructure and transparent to the fullest extent possible;
>
> o requires minimal new infrastructure;
>
> o can be implemented independently of clients in order to reduce deployment time;
>
> o can be deployed incrementally;
>
> o allows delegation of signing to third parties.

Cependant, pour qu'il soit le plus efficace possible, **il conviendrait que chaque serveur CEMTP ou SMTP dispose d'un certificat électronique l'identifiant**, et que celui-ci soit associé à la bonne entrée DNS correspondante.

Ensuite, il convient d'authentifier l'émetteur du message lui-même. Autrement dit, **qualifier de façon quasi certaine, la partie gauche de l'adresse e-mail.**

Pour ce faire, l'émission et **l'utilisation d'un certificat électronique de classe I**, c'est-à-dire prouvant l'existence de l'adresse électronique de l'émetteur, **DOIT être fortement recommandée,** voire imposée.

De plus, **il faut systématiquement utiliser, *a minima*, la fonction STMP-AUTH pour pouvoir utiliser le serveur SMTP de son domaine**. En même temps, il convient **d'éviter l'utilisation possible d'un serveur SMTP extérieur** à son propre domaine.

Néanmoins, **l'authentification par le certificat électronique de l'utilisateur sera préférable, voire obligatoire s'il dispose dudit certificat.**

De même, **la possibilité d'envoi de message sans spécifier d'adresse e-mail d'émetteur DOIT être interdite**, contrairement au protocole SMTP classique.

Dans le protocole SMTP, l'obligation d'accepter des messages ne comportant pas d'adresse e-mail d'émetteur était liée aux difficultés rencontrées avec les retours de messages d'erreurs des envois de messages.

Ces messages renvoyés par les serveurs, pour éviter des aller-retour récurrents (bouncing), ne comportent pas d'adresse email d'émission.

Cette simplicité et faiblesse a été largement utilisé par les spammeurs pour diffuser de façon importante des messages sans risquer d'être perturbés par des retours d'erreur ou des messages de rejet des utilisateurs.

Les problématiques liées au « bouncing » de messages de traitement des erreurs **DOIVENT être gérées à l'aide d'emails de types génériques associés au certificat électronique de chacun des serveurs de messagerie.**

L'intégrité du message lui-même doit être traitée, à l'aide d'empreintes (hash MD5, SHA-1 ou SHA-256) à chaque étape de la transmission du message **par chaque serveur CEMTP, au niveau même du protocole**.

Dans le cas de l'utilisation par l'émetteur d'un certificat de signature électronique, la fonction de calcul d'empreinte est intégrée au niveau de l'envoi du message. Aussi, le serveur CEMTP n'a besoin que de vérifier ladite empreinte à chaque étape pour en garantir l'intégrité.

Néanmoins, **il convient d'ajouter une empreinte complémentaire, dans un champ spécifique des en-têtes qui représentera le calcul fait sur l'ensemble des données ajoutées : Horodatage, adresse IP du serveur et son nom DNS, sa clé DKIM et l'adresse IP du serveur suivant.**

Le serveur suivant agira de même et son calcul d'empreinte intégrera également la valeur de l'empreinte du serveur précédent.

Dans le cas où l'émetteur n'utilise pas de certificat de signature électronique, le serveur d'envoi CEMTP (le MSA), intégré dans le client de messagerie, DOIT calculer l'empreinte du message avant sa transmission au serveur CEMTP.

En réception, le client de messagerie du destinataire vérifie automatiquement la validité de l'empreinte reçue et peut ainsi garantir que le message n'a pas été altéré lors de son transfert ou de sa réception.

Idéalement, **l'empreinte du message, intégrant, bien entendu, l'adresse e-mail de l'émetteur et la date d'envoi (si possible certifiée par un horodatage fiable), devrait être utilisée comme identifiant unique du message** (Message-ID = UID), facilitant ainsi, aussi bien la transmission de l'empreinte que l'identification du message par les MDAs (serveurs POP ou IMAP).

Ces derniers éléments assureront ainsi la traçabilité du message.

Concernant les aspects liés à la confidentialité des données, la sécurisation du lien entre chaque relais de messagerie permet de s'affranchir des problèmes de confidentialité des données lors de la transmission du message.

Les autres points liés à la confidentialité des données sont gérés par les MDAs et les MUAs.

Enfin, la conservation et l'archivage des logs (traces) doivent être réalisés de façon sécurisée, en signant chacun d'entre eux avant archivage, de préférence au jour le jour.

Les conditions de conservation des logs seront dépendantes des réglementation de chaque pays et des contraintes liées à la gestion des données personnelles.

En France, il convient de vérifier, le moment venu, avec la CNIL si ces traces et leur conservation doivent ou non entrer dans le cadre du « droit à l'oubli » spécifié par ses services.

5. PROPOSITIONS POUR POP ET IMAP

Le MDA (Message Delivery Agent) qui est utilisé pour stocker les messages dans l'attente de leur téléchargement ou consultation par l'utilisateur via son MUA (Message User Agent), doit respecter les mêmes contraintes et critères que ceux décrits dans le protocole CEMTP et dans le chapitre VII, savoir :

- Sécurisation des échanges
- Authentification de l'émetteur
- Intégrité des messages de bout en bout
- Traçabilité
- Confidentialité des données
- Archivage des logs (traces) de façon sécurisée

Ces évolutions du protocole permettront d'obtenir une nouvelle version CEPOP, à l'image de CEMTP, pour améliorer la confiance.

La sécurisation des échanges avec le MUA (poste client) doit respecter les contraintes de l'établissement d'un lien TLS.

Dans le cas des MDAs, l'identification de la source n'a pas lieu d'être et se trouve intégrée dans l'authentification réciproque entre le MDA et la MUA à l'aide des certificats de signature électronique.

Cette même authentification de l'utilisateur remplacera le couple « login-password » pour permettre à celui-ci d'accéder à ses données.

L'intégrité des messages ayant été suivie par les MTAs CEMTP, le MDAs n'aura qu'à vérifier le calcul d'empreinte pour valider cette intégrité.

La traçabilité sera traitée de la même manière que pour les MTAs, en ajoutant dans les en-têtes un horodatage fiable et une empreinte qui validera le moment où l'utilisateur aura téléchargé ou consulté son message.

Cette fonction permettra également d'ajouter facilement une réelle opération d'accusé de réception ayant une valeur probante tant que les systèmes d'horodatage en jeu et les certificats des serveurs auront toute leur validité, sans avoir besoin de mettre en place une infrastructure lourde.

La confidentialité des données stockées sur le MDA est un ajout important mais essentiel afin d'éviter une consultation possible des messages par un tiers.

La méthode proposée et recommandée s'appuie une fois encore sur le certificat de signature électronique simple de l'utilisateur.

En effet, il devient simple pour le MDA recevant le message d'accéder au serveur LDAP de l'Autorité de Certification afin d'obtenir la clé publique de l'utilisateur propriétaire de la boîte aux lettres.

A partir de cette clé, le MDA peut aisément crypter le message à destination unique du propriétaire de la clé privée associée.
En conséquence, seul le destinataire pourra accéder à sa boîte de messagerie et décoder ses messages pour les télécharger.

Si l'on considère que ce mécanisme est trop lourd en termes de ressources pour le MDA, il est possible de conjuguer les bénéfices du cryptage symétrique avec le cryptage asymétrique, en cryptant une clé symétrique avec la clé publique de l'utilisateur afin qu'il puisse en disposer, puis de crypter les données sur le MDA. Le choix de la clé symétrique peut être initié à la demande de l'utilisateur qui cryptera celle-ci avec la clé publique du MDA.

Un identifiant unique permet de différencier facilement les messages. Selon les conventions, il est recommandé que celui-ci soit composé du nom de domaine de l'émetteur et d'une valeur dérivée de la date.

Néanmoins, même si cette utilisation reste possible, je recommande de modifier la structure de l'UID (et de la fonction UIDL) en utilisant **l'empreinte du message** (le hash SHA-1, ou même un simple MD5), accompagné, le cas échéant d'un numéro séquentiel qui pourrait faciliter la tâche du protocole CEPOP lors de la réception des messages afin que le MUA puisse facilement vérifier la présence d'un message déjà téléchargé et d'en vérifier les séquences.

Comme pour le protocole CEMTP, le MDA se doit d'organiser un archivage sécurisé de ses logs (traces) incluant l'horodatage des connexions et téléchargement des messages par le MUA.

Bien entendu, cet archivage sera signé par le MDA afin d'apporter un niveau de preuve suffisant.

Les contraintes de conservation liées aux données personnelles sont les mêmes que pour le protocole CEMTP détaillées au chapitre précédant.

6. DES RFC COMPLEMENTAIRES ?

Le besoin de confiance dans la messagerie, en particulier, est de plus en plus présent au niveau mondial.

Ceci se retrouve dans les différentes demandes de RFC en cours (plus de 2.000 actuellement) dont une quarantaine traite de la sécurisation de l'e-mail, de la gestion des certificats de signature électronique de type X509 ou du « spam ».

Le détail de ces demandes de RFC est joint en Annexe N°2.

Par ailleurs, certains RFCs seront impactés par les propositions de modifications des serveurs CEMTP (SMTP), CEPOP ou IMAP.

En particulier les suivants :

- RFC 5322 : Internet Message Format (pour intégrer le format du Message-ID et la traçabilité)
- RFC 4409 : Message Submission for Mail : Dérivé de SMTP pour gérer des options locales de sécurité à l'émetteur et l'envoi uniquement au premier serveur SMTP.
- RFC 3461 : SMTP Services Extensions
- RFC 2554 : SMTP Service Extension for Authentication
- RFC 3207 : SMTP Service Extension for Secure SMTP over Transport
- RFC 2822 : Internet Message Format, pour les formats des en-têtes et des dates
- RFC 5322 : Format des en-têtes de messages

Par ailleurs, certains nouveaux RFC pourraient être proposés ou adaptés pour l'archivage et la confidentialité.

1. Archivage

Pour l'archivage des messages électroniques, qu'il soit réalisé au niveau du poste de travail ou sur le serveur de messagerie (POP ou IMAP (MDA)), il y a lieu de définir un RFC qui respecte les caractéristiques normalisées de la NFZ42013 puis ISO CN171), tout en suivant les contraintes de la CNIL quant à la durée de vie des archives, le droit à l'oubli et la confidentialité.

Bien entendu l'archivage des logs des serveurs CEMTP et des MDAs devra suivre ces normes et standards et sera adapté lors de la publication de tels RFC.

Ce sujet pourrait à lui seul justifier un rapport à part entière, mais comme il va bien au-delà du domaine de la présente thèse, il ne sera pas traité ici. Néanmoins, ce thème pourrait en être une prolongation utile.

2. Confidentialité

Concernant la confidentialité des données, il convient de séparer plusieurs cas classifiés de la façon suivante :

- **Confidentialité des données pendant le transport du message.**

  Cette option peut, et doit, être traitée par l'établissement d'un lien sécurisé de type TLS entre le client de messagerie émetteur, chacun des relais de messagerie et ce jusqu'au MDA final.
  Il n'est pas nécessaire d'ajouter une complexité supérieure à ce lien TLS tant que celui-ci sera réputé fiable.

  De plus, si l'émetteur le souhaite, il faut rappeler qu'il peut crypter un message spécifiquement pour son destinataire, mais cela reste de sa responsabilité et ne doit pas interférer au niveau des protocoles.

- **Confidentialité des messages stockés sur les serveurs (MDAs).**

  Ce point est délicat à gérer, car il convient de disposer d'un mécanisme le plus simple possible tout en étant fiable pour une éventuelle récupération du mot de passe ou de la clé ayant servie à rendre les données confidentielles pour toute personne en-dehors de l'utilisateur de ladite messagerie.

La meilleure solution, qui sera implémentée dans la maquette proposée, est l'utilisation de la clé privée du certificat électronique de l'utilisateur (de préférence sous son contrôle exclusif).
Cette option permet une certaine transparence pour l'utilisateur, mais implique **une certaine complexité lors du renouvellement des certificats de signature électronique, car il est nécessaire de décrypter l'ensemble des données avec l'ancienne clé, pour les crypter, de nouveau, avec la nouvelle clé.**

Ceci pourrait être réalisé de façon automatisé à la demande de l'utilisateur, à l'aide d'un programme spécialement développé à cet usage.

L'option consistant à utiliser un mot de passe, même protégé par des échanges sécurisés, comporte un risque de conservation dudit mot de passe sur le serveur de messagerie lors du cryptage des données. Or si le serveur est « corrompu », le mot de passe risque d'être capturé et utilisé pour décoder les données.

En revanche, le mécanisme de récupération du mot de passe est plus simple à mettre en œuvre et plus pérenne pour le long terme.

- **Confidentialité du contenu vis-à-vis des tiers** (message crypté à destination d'un seul récepteur)

Cette partie est pratiquement la plus simple à traiter, car il existe déjà de nombreux documents traitant du sujet, en particulier les RFC suivants : RFC 1421 (PEM) / 4902 (OPES) / 3156 (Open PGP) / 3852 (Crypto Message Syntax).

De plus, l'utilisation d'un certificat de signature électronique, même de Classe 1, permet, si l'Autorité de Certification le souhaite, de traiter la confidentialité des messages électroniques.

La seule difficulté de cette option réside dans l'éventuelle récupération d'une clé privée qui aurait été perdue ou révoquée sans que l'utilisateur n'ait eu le temps de décrypter ses données puis de les crypter, de nouveau, avec un autre certificat.

L'autre option qui consiste à utiliser une cryptographie symétrique, permet plus aisément de gérer le « key recovery ».

Plus précisément**, la conservation par l'utilisateur du mot de passe ou de la clé symétrique utilisée peut être réalisée avec une cryptographie asymétrique en utilisant sa clé privé de signature électronique pour conserver son (ou ses) mot(s) de passe(s).**

Par ailleurs, la clé de cryptographie symétrique peut être conservée de façon fiable et pendant une longue durée par l'AC.

Il convient de noter que le problème du « key recovery » ne se pose que dans l'hypothèse où le destinataire conserve sur son ordinateur la version cryptée du message ou du document, et non sa version décryptée, ce dernier cas étant le plus fréquent, puisque l'objectif de l'envoi crypté dans ce mode est bien la confidentialité de bout en bout, mais pas la conservation sur le long terme de ces documents sous forme cryptée.

A ce titre, une protection contractuelle peut être proposée par l'AC pour ne conserver la possibilité de « key recovery » que pendant une période définie.

- **Protection des données personnelles** (usage des e-mails pour le « spam » en particulier)

  Le point lié à la protection des données personnelles, et au respect des contraintes imposées par la CNIL (pour la France), trouve déjà une première réponse dans la sécurisation des données pendant le transport, qui permet d'éviter « l'écoute » trop simple des paquets sur le réseau pour y saisir des données comme les adresses e-mail.

  Les autres aspects de cette protection nécessitent une étude à part entière pour envisager toutes les options incluant la durée de vie des messages, le droit à l'oubli qui devrait être compris par l'ensemble des utilisateurs, la non-diffusion de données « marquées » comme privées…

IX. Le modèle de confiance

La mise en place de ces recommandations et propositions de modifications des protocoles doit se faire dans le cadre d'une maquette opérationnelle permettant de montrer le bien fondé et l'adéquation de ces mécanismes.

Pour ce faire, il est nécessaire de détailler les principes de fonctionnement de ce modèle.

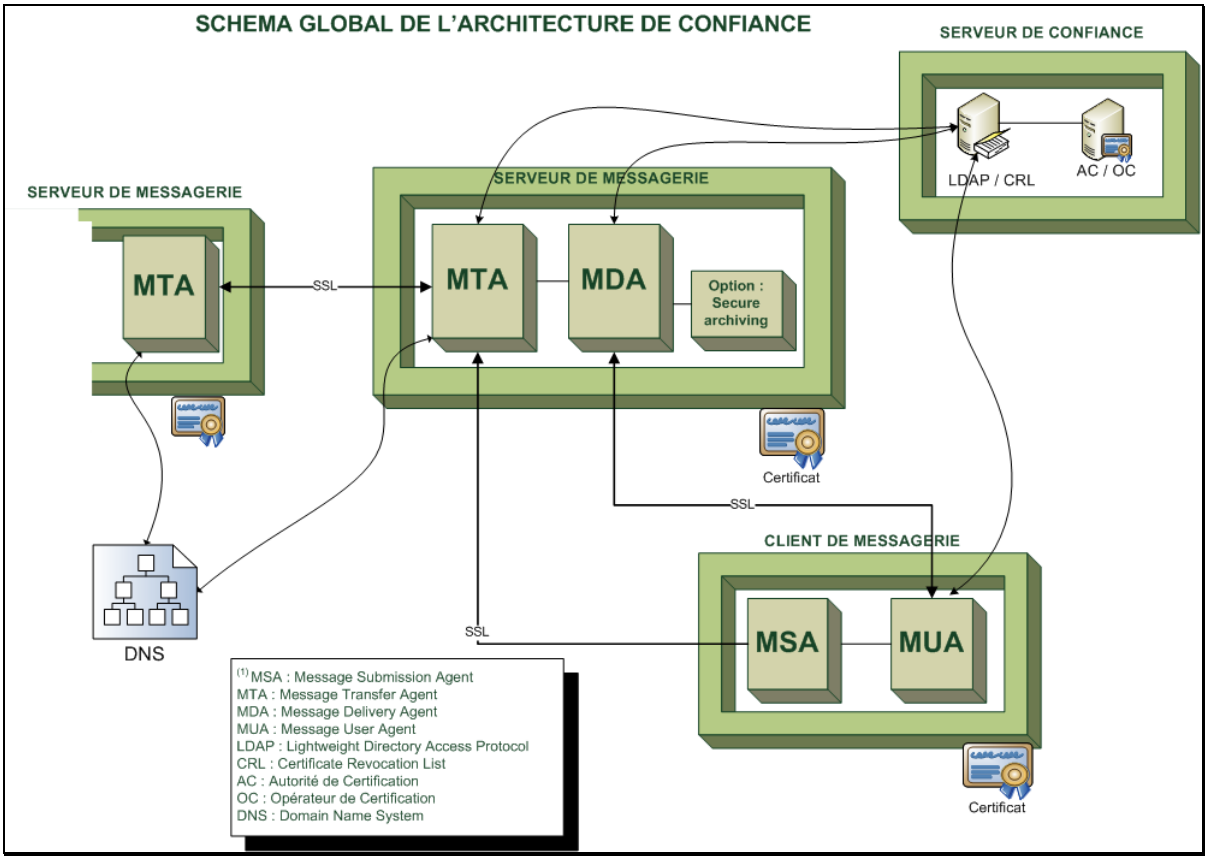
1. STRUCTURE DE L'ARCHITECTURE DE CONFIANCE

Le principe général est de **réaliser une maquette d'un ensemble stable** mais évolutif intégrant le nouveau protocole CEMTP et les recommandations prévues dans cette thèse pour les MDAs et les clients de messagerie.

De même, le pilote devra intégrer la mise en œuvre du serveur de confiance apte à délivrer de façon sécurisée et automatisée les certificats de classe I qui seront utilisés par le logiciel client de messagerie, lui-même développé dans le cadre de cette maquette.

Ce logiciel client utilisera le principe proposé dans la RFC 4409 concernant un agent de liaison avec le serveur CEMTP (MTA), nommé MSA (Message Submission Agent).

Le schéma suivant décrit de façon globale cette maquette :



Pour simplifier le système à mettre en place, pour le transfert et la réception des messages, seuls les protocoles CEMTP et CEPOP ou POP3S seront implémentés.

Plusieurs étapes sont nécessaires pour décrire le fonctionnement du système, de la délivrance du certificat de signature électronique de classe I aux échanges entre les différents agents de messagerie (MSA, MTA, MDA, MUA).

Il faut rappeler qu'un certificat de signature électronique de classe I n'authentifie qu'une adresse e-mail donnée et non pas l'identité civile de l'utilisateur.

En revanche, les certificats utilisés par les serveurs de messagerie sont du même type que ceux utilisés par les serveurs web pour établir des liens sécurisés en TLS.

Ils sont donc délivrés de façon « traditionnelle » avec les contraintes de validation que cela implique, en particulier l'envoi de documents relatifs aux personnes et à la société qui les exploite, et l'éventuelle délivrance en face à face pour en garantir la valeur.

De nombreuses organisations référencées comme PSCE (Prestataires de Services de Certification Electronique) sont à même de distribuer ces certificats pour les serveurs.

2. SCHEMA GLOBAL DES ECHANGES

1. Délivrance du certificat de signature électronique :

Cette opération est réalisée comme suit :

L'utilisateur envoie une demande à l'Autorité de Certification (AC) pour obtenir un certificat de classe I.
L'AC vérifie la réalité de ladite adresse e-mail par un échange avec l'utilisateur.

Une fois cette validation effectuée, l'AC demande à l'OC de générer le certificat.

L'OC envoie alors un e-mail à l'adresse de l'utilisateur avec un lien sécurisé afin qu'il puisse télécharger sont certificat, constitué de sa clé privée et de sa clé publique, avec les explications nécessaires à son installation et son utilisation.
Il est important à ce stade, que les mécanismes d'installation et d'utilisation soient le plus transparents pour l'utilisateur final. Une interface spécifique permettant cette simplification sera intégrée dans le logiciel de messagerie client (MUA).

De même une interopérabilité entre les différents logiciels et navigateurs doit être assurée, et, à cet effet, un logiciel d'installation automatisé proposé par l'OC permettra de vérifier la bonne configuration des différents « magasins » et logiciels associés.

A la suite de la validation de l'installation sur le poste de travail de l'utilisateur, l'OC publiera la clé publique de l'utilisateur dans le serveur d'annuaire LDAP qui sera ainsi disponible, soit publiquement, soit de façon restreinte au choix de l'utilisateur.

Plusieurs opérateurs proposent déjà la délivrance de certificats de classe I gratuits tels que :

CAcert.org : http://www.cacert.org
StartComm : http://www.startssl.com

Bien entendu, ces certificats peuvent être utilisés, ainsi que tous les certificats de signature électronique diffusés par l'ensemble des opérateurs de confiance internationaux.

Une interface spécifique assurant cette interopérabilité entre les différents types de certificats fera également partie du projet.

2. Echanges entre MSA, MTA et MTA :

Les échanges entre les MSA (Message Submission Agent) et les MTA (Message Transmission Agent), et entre les différents MTA successifs peut se décomposer comme suit :

Une requête est envoyée par le MSA au MTA sur le port adéquat.

En réponse, le MTA établit un lien TLS avec le MSA afin de sécuriser le lien.
Le MTA vérifie, dans l'annuaire LDAP de l'Autorité de Certification, l'existence et la validité du certificat présenté par le MSA.

Si celui-ci est valide, alors le MTA met le message proposé en file d'attente (queue).
Le MTA procède à l'analyse des destinataires :
Il vérifie la présence de l'adresse électronique dans l'annuaire LDAP de l'AC.
Si le certificat est présent et valide, alors le message est transféré directement à son serveur de messagerie dont les coordonnées seront enregistrées au sein de l'annuaire ou du certificat.
Si l'adresse n'est pas incluse dans l'annuaire, le MTA va interroger le serveur MTA du domaine du destinataire.
Si celle-ci n'existe pas, alors il retourne un message d'erreur (signé) à l'émetteur.
Si l'adresse existe, alors le MTA établit un lien TLS avec le MTA du destinataire. Dans le cas où le MTA du destinataire ne peut être joint directement, le lien sera un « relais » vers le MTA final.
Le MTA émetteur ajoute alors ses coordonnées (adresse IP et FQDN), les données d'horodatage fiable à partir de son serveur d'horodatage interne, rajoute sa signature puis calcule l'empreinte de l'ensemble de ces données rajoutées qu'il enregistre dans un champ d'en-tête spécifique « CEMTP-HASH ».
Enfin, il transmet le message au MTA suivant et enregistre les données de « log » dans son fichier sécurisé.

Le MTA, en réception, commence par mettre en file d'attente le message.
Il recalcule les empreintes afin de vérifier la bonne transmission des données.

Si ce calcul n'est pas correct, il envoi un message au MTA d'émission afin qu'il retransmette ledit message.
Si les empreintes sont correctes, il retourne un acquittement au MTA d'émission pour clore le lien TLS.
De même, le MTA d'émission ayant reçu l'acquittement, ajoute cette information dans le fichier de « log ».

3. Echanges entre MTA et MDA :

A la suite des échanges décrits ci-dessus entre les MTA, et en bout de la chaîne de transmission, le MTA de réception transmet, localement, le message au MDA (Message Delivery Agent).

Si le MDA est distant, même dans le cas d'ordinateurs différents situés côte à côte, alors l'établissement d'un lien sécurisé de type TLS est obligatoire.

Le MTA envoi une requête sur le port adéquat du MDA qui possède un processus d'écoute permanent.

A la réception du message, le MDA vérifie si le message est signé par l'émetteur.
Si c'est le cas, il se connecte sur l'annuaire LDAP de l'AC afin d'en vérifier la validité.
Sinon, il envoie une requête au MTA de l'émetteur afin de valider l'adresse électronique de celui-ci.
En cas d'échec des vérifications ou d'absence d'adresse e-mail de l'émetteur, le message sera systématiquement rejeté.

<u>REMARQUE</u> : Le processus d'analyse de la validité d'un émetteur est sensiblement plus complexe, mais seule une version simplifiée est présentée ici.

Outre les vérifications effectuées sur l'émetteur, le MDA évaluera un niveau de probabilité de « spam » qui sera qualifié selon des mesures statistiques pointues telles qu'évoquées au chapitre X.4.

Ce poids sera comparé au seuil défini par l'utilisateur lui-même afin, soit de rejeter immédiatement le message, soit de le marquer et de le stocker dans le MDA.

le MDA analyse également l'état du « compte » du propriétaire de la messagerie électronique du destinataire.
Si cet état est invalide (compte « périmé », boîte saturée…) il renvoie un message au MTA local avec un code d'erreur et un message « clair » adapté. Ce dernier MTA relaye alors ce message d'erreur vers l'émetteur du message.

Si l'état du compte est valide, le MDA vérifie les empreintes du messages, et le cas échéant redemande la transmission dudit message.

Dès que tous les paramètres ont été validés, alors le MDA acquitte le MTA local avec la validation de son certificat électronique et clos l'échange avec ledit MTA.
Le MTA local recevant cet acquittement, peut, si l'option d'accusé de délivrance a été activée, retourner un message comportant l'acquittement de la réception par le MDA avec l'ensemble des signatures nécessaires.

ATTENTION : Cet accusé de délivrance N'EST PAS, *a priori,* un accusé de réception au sens légal, car le destinataire n'a pas encore reçu le message sur son ordinateur, ni ne l'a consulté.
Cette étape pourra être réalisée, le cas échéant, lors de la consultation ou du téléchargement du message par le MUA.

Le MDA accède alors à l'annuaire LDAP de l'AC afin d'obtenir la clé publique de l'utilisateur destinataire.
Avec cette clé publique, il crypte les données du message sur le disque.

Ensuite il met à jour sa base de données pour tenir compte de l'arrivée de ce nouveau message.
De même, il enregistre dans ses « logs » sécurisés les informations relatives aux échanges avec le MTA et l'enregistrement du message.

4. Echanges entre MUA et MDA :

Echanges MUA et MDA

Développer la vérification de sigelec et de hash à chaque étape, ainsi que l'ajout « certifié » de chaque MTA.

3. STRUCTURE DES SYSTEMES

Dans ce principe, nous avons du côté opérationnel deux serveurs et le programme client de messagerie :

- MTA Serveur CEMTP sécurisé
- MDA Serveur CEPOP sécurisé
- MUA client CEMTP et CEPOP

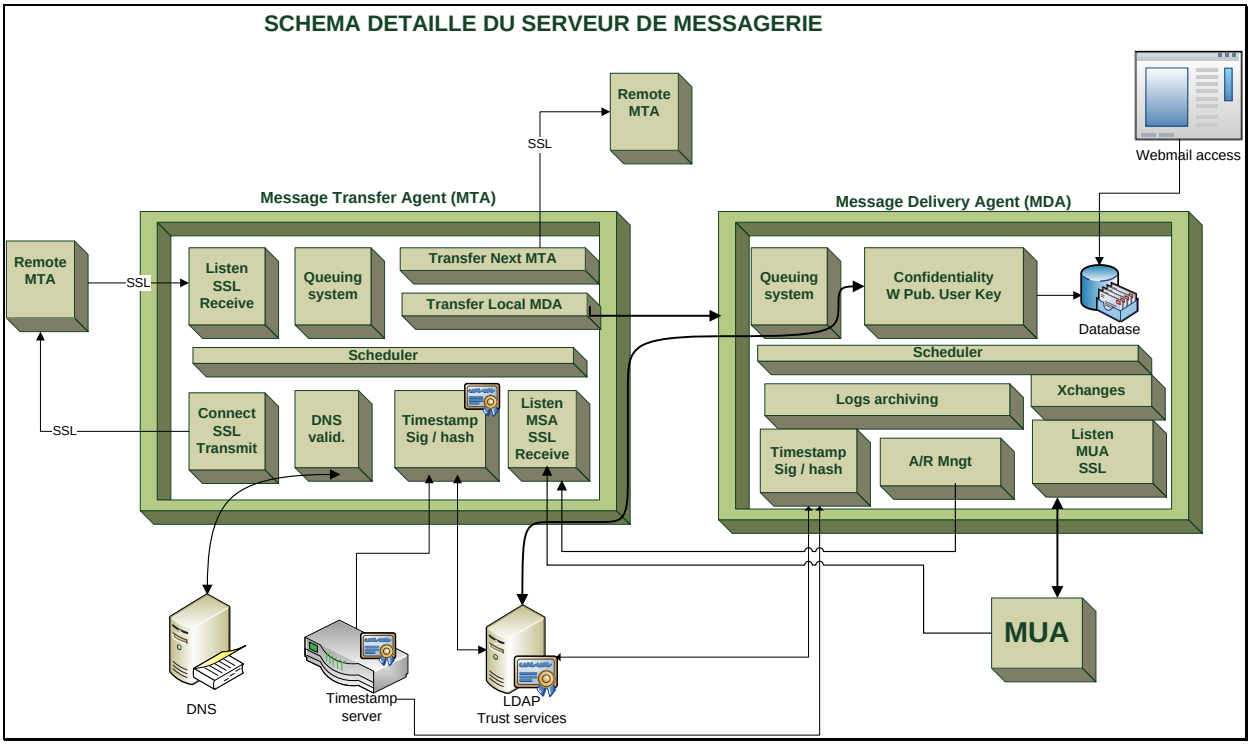
Par ailleurs, il y aura un gestionnaire global de la PKI « pkiemail.com » qui aura en charge la génération, semi-automatisée, de délivrance des certificats.

Ce même serveur aura donc les trois fonctions d'Autorité de Certification, d'Autorité d'Enregistrement et d'Opérateur de Certification pour les utilisateurs (e-mails).

Les certificats des serveurs MDA et MTA, ainsi que celui de l'AC seront des certificats traditionnels de serveurs acquis auprès d'une Autorité de Confiance référencée, en particulier dans les navigateurs et clients de messagerie traditionnels.

A côté de ce serveur de confiance, un serveur LDAP gérant l'ensemble des certificats et des listes de révocation (CRL) sera mis en place. Il devra évoluer vers une configuration hiérarchique en fonction de l'évolution de la demande.

La structure détaillée des serveurs de messagerie (MTA et MDA) peut être représentée de la façon suivante :

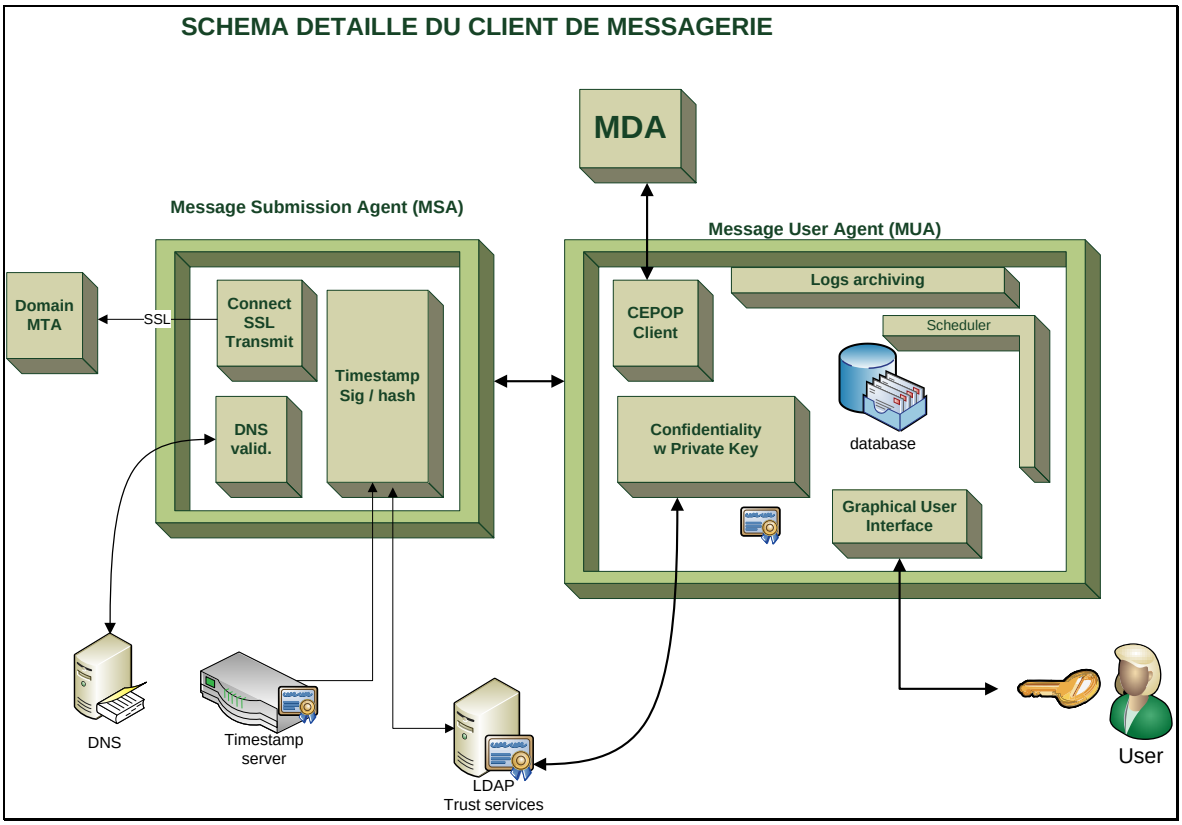


Du côté du client, le MUA, le programme est divisé en plusieurs parties autonomes et disposant d'une base de données centralisée :

- Le MSA gère la file d'attente des envois,
- Le client CEPOP gère les accès au MDA,
- L'interface homme/machine accède aux données et gère l'interaction utilisateur,
- La base de données elle-même qui gère les priorités et les blocages d'accès aux données,

L'ensemble est coordonnée par un gestionnaire / scheduler qui optimise les temps d'accès aux serveurs et aux données.

Le schéma de ce client de messagerie peut être représenté ainsi :



4. COMPATIBILITE ASCENDANTE

La difficulté va provenir principalement de la compatibilité ascendante nécessaire pour permettre le dialogue avec les serveurs SMTP existants ainsi qu'avec les MDAs existants.

Cette compatibilité va réduire sensiblement l'effet de sécurisation, d'identification et de traçabilité lors des passages par les relais SMTP.

Néanmoins, le serveur CEMTP, et la structure de fonctionnement de ses caractéristiques, notamment liées au format des messages, sera organisé de façon à transiter de manière transparente vis à vis des serveurs SMTP classiques.

En conséquence, si un message passe d'un serveur CEMTP à un serveur SMTP, puis de nouveau à un serveur CEMTP, l'essentiel des fonctionnalités se propagera, et seules la confidentialité et la traçabilité de l'étape SMTP ne seront pas garanties.

La compatibilité sera encore plus critique vis-à-vis du MUA pour le nouveau logiciel adapté aux protocoles CEMTP et CEPOP, qui devra pouvoir également communiquer avec les serveurs SMTP et POP3 classiques.

Il conviendra d'y ajouter les alertes nécessaires lors des connexions moins sécurisées avec les serveurs existants.

De même, la connexion d'un MUA classique sur un serveur CEMTP et CEPOP devra être possible avec un certain nombre de limitations liées aux conditions de fonctionnement de ces protocoles.

X. La mise en œuvre du modèle

La mise en œuvre du modèle technique détaillé aux chapitres précédents devra pouvoir se faire avec une transition la plus transparente possible pour l'utilisateur, ainsi que pour les opérateurs choisissant de basculer sur ce mode de sécurisation.

1. UN CLIENT OPEN SOURCE

Bien entendu, pour déployer une maquette de façon suffisante pour avoir un retour sérieux, il convient de fournir un client « open source » apportant de réelles nouveautés pour pouvoir faire « basculer » les utilisateurs vers ce nouvel outil en insistant sur les apports sécuritaires de cette solution.

L'évolution du nombre d'utilisateurs (MUA) de cette solution favorisera le déploiement des serveurs adaptés.

Pour faciliter le développement de cette solution, on peut, au départ, se baser sur le projet « trustedbird » par exemple, initié par la DGA sur la base de la plateforme Thunderbird de la fondation Mozilla afin d'y intégrer les aspects de sécurisation, de gestion des certificats, de confidentialité, des accusés de réception…

Le projet « trustedbird » va dans le même sens que la présente thèse pour développer la confiance dans la messagerie électronique, mais en se basant uniquement sur le client de messagerie.

En ajoutant les apports techniques liés à la modification des protocoles CEMTP et CEPOP, il convient d'y ajouter les fonctionnalités complémentaires adaptées.

Précisément, mettre en œuvre un MSA (Message Submission Agent) intégré au logiciel, mais fonctionnant de façon indépendante et un client CEPOP fonctionnant également comme un processus indépendant, le tout associé à une base de données structuré pour stocker les messages avec les options de confidentialité, feront partie des premières améliorations de ce logiciel.

2. UN SERVEUR ADAPTE POUR LES ISP

A partir de cette maquette, il conviendra de développer des serveurs CEMTP ainsi que des serveurs CEPOP de qualité professionnelle à destination des ISP afin d'accélérer le déploiement de ce protocole et la sécurisation des messages électroniques.

Ce que l'on appelle « scalability », c'est-à-dire la capacité d'évolution en puissance d'un serveur en fonction de l'évolution de la demande, est essentielle pour des fournisseurs d'accès ou de services de messagerie.

La répartition de la puissance entre différents serveurs, ou différents clusters devra s'accompagner de modules de surveillance et de contrôle des opérations de chacun d'entre eux.

De même, l'organisation du serveur de confiance, capable de délivrer de nombreux certificats de classe I et les liaisons avec l'annuaire LDAP, impliquera une nouvelle répartition du rôle et une structure hierarchisée d'accès à l'annuaire.

Celle-ci pourrait ressembler au fonctionnement du DNS avec une répartition géographique des serveurs de niveau inférieur afin d'optimiser le trafic Internet lors des requêtes qui deviendront de plus en plus fréquentes.

La contrepartie de la sécurité des échanges est une augmentation du nombre des requêtes vis-à-vis de certains serveurs lors de l'envoi de chaque message.

3. EXTENSIONS ET SERVICES

Sur ces bases, des extensions seront mises en place afin d'intégrer, aussi bien au niveau de l'Autorité de Certification que des serveurs CEMTP, des MDAs et des clients de messagerie, **une interopérabilité totale entre les différentes catégories de certificats existants sur le marché, qu'ils soient commerciaux ou gratuits, basés sur X509 ou sur PGP.**

Cette interopérabilité est un élément clé du développement de cette solution et de la confiance qui y est associée.
L'utilisation des certificats existants et de ceux issus d'Autorité de Confiance extérieures doit être une réalité.
Il sera possible à cet effet de développer une forme de label ou d'agrément basé sur des critères de déontologie de l'émission de certificats qui permettrait d'évaluer un niveau de garantie offert par ces Autorités de Confiance.

Les fonctions d'archivage sécurisé devraient être externalisées auprès d'opérateurs de confiance existants.
Néanmoins, il sera également possible d'assurer au niveau des fournisseurs de messagerie, un service d'archivage utilisable aussi bien pour les besoins internes (logs) que pour des services fournis aux utilisateurs dans le but d'archiver de façon sécurisée leur messages.

A ce titre, les services « EHE » (Exchange Hosted Encryption) et « Exchange OnLine Archiving » de Microsoft peuvent servir de modèle.

D'autres services comme l'émission de Certificats de Classe 2 ou des services d'interopérabilité des certificats pour les PSCE pourront être envisagés.

4. SUIVI DU FONCTIONNEMENT ET DU « SPAM »

Les statistiques actuelles du trafic de la messagerie et de celles du « spam » sont littéralement astronomiques : Les « spams » qui finissent par atteindre leur cible sont aujourd'hui de l'ordre de la centaine, par jour, et par internaute.

Ceci montre à l'évidence que les filtres mis en place sont encore insuffisants, et sujets à caution : Il y a lieu de les renforcer par d'autres méthodes qui viendront compléter ce qui existe déjà.

▪ Les applications possibles des modélisations stochastiques au contrôle des spams :

Dès lors, et compte tenu des chiffres actuels qui reflètent ce phénomène moderne et désastreux des « spams », une approche statistique et probabiliste pourrait à l'avenir se trouver bien adapté à ce problème, pour diverses raisons, telles que :

- Analyses des **caractéristiques monodimensionnelles** des nuages de points représentant les « spams » ;

-Identification probabiliste des « spammeurs » ;

- **Processus bayesiens** par étages successifs de l'identification des « spams » dans un ensemble de mails destinés à un utilisateur final (cette méthode, déjà utilisée par les logiciels anti-spam doit être étendue.) ;

- Calcul des **liaisons** pouvant exister entre une caractéristique de « spam » (par exemple ceux liés à un hameçonnage), et une caractéristique liée à l'utilisateur (par exemple l'appartenance à un fournisseur d'accès à internet, ou une localisation géographique, ou une période particulière, par exemple dans les 15 jours de l'inscription d'un nouvel utilisateur à un fournisseur d'accès) : Sur la base de ces couples de caractéristiques, il devient possible d'étudier leur coefficient de corrélation ou leur matrice de contingence : Ce type d'étude concernera essentiellement les **analyses bi dimensionnelles** des « spams » :

- Modélisation stochastique **causale** de la création, et de la diffusion des « spams », permettant, si les modèles sont qualitativement bien ciblés et de bonne qualité, de diagnostiquer leur existence potentielle ou réelle, et de prévenir leur distribution : Les modèles en question sont des modèles de type **ANOVA** (**AN**alysis **O**f **VA**riance) ;

- Modélisations stochastiques **temporelles** de la création et de la diffusion des « spams », basés sur les **Processus Stochastiques ;**

- Modélisations stochastiques d'**Analyse Factorielle** de recherche des facteurs importants dans la lutte contre les « spams ».

- **LES PRINCIPES D'ÉTUDE DES NUAGES DE POINTS « SPAMS » REFÉRENCÉS PAR LEURS CARACTÉRISTIQUES PRINCIPALES, AU TRAVERS DES MODÉLISATIONS STOCHASTIQUES :**

Le domaine des statistiques et du calcul de probabilité n'est pas le propos de la présente thèse.

Il n'est donc pas utile de développer ici même ces thèmes et d'indiquer en détails comment ils seraient mis en place sur le plan opérationnel, dans le cadre de la lutte contre les « spams ».

Néanmoins, l'utilisation des modèles stochastiques va vraisemblablement s'imposer à brève échéance dans ce cadre.

Alors, à titre de projection vers le futur, il semble utile de préciser comment ces outils statistiques pourront être mobilisés dans le cadre de la présente thèse, en formalisant de manière théoriquement souhaitable les observations et constats qui abondent expérimentalement dans le domaine du « spam ».

Le principe des modèles stochastiques de ce type est relativement simple et peut être schématisé comme suit:

Dans un premier temps, par la force de l'observation, et de l'expérimentation, il s'agit de définir, de relever, et de mesurer les caractéristiques X des « spams », telles qu'elles apparaissent aux fournisseurs d'accès à internet qui les reçoivent pour le compte de leurs clients. Certaines de ces variables seront considérées comme causales, et d'autres seront considérées comme résultantes, ou temporellement dépendantes, dans le cadre des études stochastiques à mener.

Les modèles étudiés auront alors pour objet d'analyser les liens pouvant exister entre les variables résultantes mesurées et observées (par exemple : nombre de « spams » produits pour être diffusés, typologie des « spams » créés, nature des cibles, types d'objectifs visés, etc..), et les variables causales (localisations géographiques de la production de « spams », modalités de création, modalités de diffusion, moments choisis pour la diffusion, etc..) ou temporelles observées.

Ces liens seront d'abord exprimés sous formes de fonctions paramétrables liant les variables observées, et les variables causales ou temporelles :

Ainsi, on pourra écrire **Y= F(X1, X2,X3 ; p1,p2,p3, p4 ; e)**
Dans lequel :

F est la fonction recherchée à partir des données existantes, pour ensuite être utilisée comme fonction prédictive destinée à examiner l'impact des remèdes proposés sur les caractéristiques observées ;

X1, X2,X3 sont les caractéristiques observées lors des différents constats expérimentaux ;

Y est la variable résultante également observée lors des différents constats expérimentaux ;

p1,p2,p3,p4 sont les paramètres du modèle inconnus au départ, et qu'il s'agit d'optimiser pour un modèle donné, à partir des observations et des constats effectués ;

e est l'erreur commise à chaque observation.

La somme $\sum \mathbf{e^2}$ représente alors la somme des carrés des erreurs que nous commettons au niveau des observations en acceptant le modèle choisi, et en calculant son minimum grâce à l'optimisation des paramètres adoptés ; il s'agit en gros d'une mesure des erreurs que nous commettons à cause du modèle choisi.

Par ailleurs, $\sum \mathbf{Y^2}$ représente le carré des variables résultantes observées, c'est-à-dire en gros, ce que nous sommes capables d'expliquer de nos observations, grâce au modèle choisi.

Le rapport $\sum \mathbf{e^2}/\sum \mathbf{y^2}$ représente alors la proportion de nos erreurs, par rapport à ce que nous sommes capables d'expliquer, dans le cadre du modèle étudié, et avec les constats effectués.

Ce rapport, appelé **« goodness of fit »**, ce qui peut se traduire par « qualité du modèle », devient pour nous une mesure de la qualité du modèle que nous étudions.
Si ce rapport est faible, alors le modèle étudié est bon, et pourra être utilisé à titre prédictif pour projeter dans le futur les caractéristiques probables des « spams » dont nous voulons contrôler le devenir grâce au modèle proposé.

Il est constant qu'une approximation linéaire facilitera considérablement l'étude statistique des modèles de ce type, au travers de l'**ANOVA** (**AN**alysis **O**f **VA**riance), des **Processus Stochastiques** (processus normaux, modèles épidémiques, calcul des tendances, et des oscillations, processus de Markov, etc..), ou de l'**Analyse Factorielle** (recherche de l'axe principal , et du centroïde du nuage de « spams » étudiés, et de leur signification, recherche des axes secondaires et de leur signification, etc..).

Il est également constant que l'analyse des liens pourra considérablement aider à améliorer le contrôle de ces « spams », à travers la recherche de ceux ayant des auteurs se connaissant entre eux, ou utilisant des techniques d'écriture voisines.

Le calcul de matrices de contingence, et de coefficients de corrélation ad hoc, pourrait permettre d'établir des liens à forte probabilité entre les auteurs et les distributeurs de ces « spams », et d'aider ainsi au contrôle par les forces de l'ordre de ces personnes.

Sans aller jusqu'à la mise en place effective d'un certain nombre de modèles stochastiques dont l'étude s'avère de plus en plus souhaitable et effective, il semble que le fait de pointer dans cette direction au niveau de la recherche théorique est tout à fait prometteur dans le domaine du spam, et susceptible d'apporter rapidement des solutions prédictives aux problèmes d'identification et de contrôle qui sont aujourd'hui mal résolus, et pourtant si cruciaux dans ce domaine.

Le principe de mise en œuvre de ces mécanismes d'analyses statistiques est de créer un indice du niveau d'évaluation du « spam » en fonction de l'ensemble des critères calculés.

Ce « poids » donnera un idée de la fiabilité du message reçu (*X-Reliable*) permettra à l'utilisateur de « régler » le niveau de fiabilité qu'il considère acceptable.

En-dessous de ce niveau, les messages seront rejetés directement par le MDA, et acceptés au-dessus.

La responsabilité du rejet sera effectivement entre les mains de l'utilisateur, du fait de son choix dans la position du « curseur » de fiabilité.

XI. Conclusion

L'utilisation grandissante de la messagerie électronique dans les échanges dématérialisés, aussi bien pour les entreprises que pour les personnes physiques, génère un manque de confiance à la fois dans les informations transmises et dans les émetteurs de ces messages.

Compte tenu du constat de l'augmentation phénoménale du « spam » allant jusqu'à 95% du trafic de la messagerie électronique, représentant 250 milliards de messages par jours, dont 50 milliards arrivent dans les boites aux lettres des utilisateurs, on comprend mieux l'exaspération qu'ils ressentent à l'égard de ce fléau et sur la perte de confiance globale dans la messagerie électronique.

En conséquence, il convient de mettre en œuvre des solutions permettant de restaurer cette confiance, et par là-même, réduire sensiblement certaines catégories de « spams ».

1. SYNTHESE DES CAUSES

Le premier travail d'analyse a porté sur les causes de cette perte de confiance dans la messagerie électronique.

Celles-ci sont dénombrables, en-dehors des aspects psychologiques qui ne sont pas pris en compte ici et sont décrites ci-après:

- Le « spam » et son augmentation qui devient critique :

  Avec 250 Mds de « spam » par jour, dont au moins 50 Mds atteignent l'utilisateur final pour ne disposer en bout de chaîne que de moins de 20% de messages utiles, le « spam » est un véritable fléau.

- L'absence d'identification des sources des messages

  La précision de l'origine de l'envoi des messages est très faible actuellement, et le « spoofing » en développement.

- Le manque d'authentification des émetteurs de messages

  L'émetteur d'un message n'est pas authentifié, et n'est parfois même pas indiqué dans les en-têtes. Ce facteur aide au développement du « spam » et plus généralement de l'anonymat.

- Absence de sécurisation des liens

  Les échanges entre les serveurs et avec les clients de messagerie sont réalisés le plus souvent en clair, laissant la place à des interceptions de ceux-ci et au manque de confidentialité.

- Pas de mécanisme assurant l'intégrité des messages

  On ne peut garantir le contenu d'un message, ni la bonne qualité de sa transmission. De fait, plusieurs expertises portent sur la validité d'un message retransmis et modifié pour tromper l'environnement.

- Absence d'un horodatage fiable et de traçabilité sérieuse

  Les dates et heures, ainsi que les données de transit entre les différents serveurs, ne peuvent être garanties et sont donc sujettes à caution.

L'ensemble de ces causes impactent également le mécanisme de transport (SMTP) et les serveurs de messagerie (Serveurs POP ou IMAP) utilisés par les fournisseurs de services, ou directement par les entreprises.

La somme de ces causes entraîne directement une perte de confiance dans la messagerie électronique dans son ensemble.

Or, au regard des aspects techniques de ces faiblesses, il est possible d'y apporter des solutions.

2. SYNTHESE DES SOLUTIONS

Un méthode envisageable pour répondre à ces faiblesses consiste à n'échanger que dans un « réseau fermé d'utilisateurs » et de se servir des couches supplémentaires fournies, en général, dans des solutions propriétaires onéreuses.

Certaines sociétés vont même jusqu'à rejeter la messagerie électronique en se tournant vers d'autres moyens de communication.

En analysant de façon plus approfondie ces causes, on constate qu'elles proviennent pour beaucoup de faiblesses ou de limitations des protocoles standard de l'internet que sont SMTP et POP3.

Même si, par principe, des organisations comme l'IETF ne souhaitent pas faire évoluer ces protocoles pour leur en garder leur simplicité, il faut considérer que l'évolution du trafic, le développement du « spam » et le besoin de disposer de solution de sécurisation doit l'emporter sur des critères conservateurs.

Aussi, la modification des protocoles fondamentaux est une solution qui permet d'apporter des réponses sans alourdir le fonctionnement des serveurs et des clients de messagerie.

A ce titre, la rédaction d'un nouveau protocole de transfert des messages, extension du protocole SMTP, appelé **« CEMTP » (CErtified Mail Transport Protocol) et dont les caractéristiques sont définies dans cette thèse**, est un point de départ essentiel.

Ces améliorations nécessitent le développement de **l'utilisation du certificat de signature électronique simple (classe I) authentifiant l'adresse e-mail** de l'émetteur, ce qui permet de garantir l'authentification de l'émetteur, l'intégrité des données reçues et la sécurisation de l'accès aux données.

Cette fonction est un point essentiel de la lutte contre le « spam » en amont de la réception de celui-ci.

L'amélioration de la **traçabilité des messages électroniques, et l'ajout de fonctions d'horodatage fiable** permettront également d'apporter une confiance supérieure.

L'intégrité des données est assurée par l'utilisation du certificat de signature électronique, mais même dans le cas où un certificat n'est pas utilisé, **le protocole assurera une garantie d'intégrité des données transmises de bout en bout.**

La mise en place des **mécanismes de confidentialité**, tant au niveau des échanges entre les serveurs de messagerie (MTA) qu'au niveau du stockage des données réalisé chez les opérateurs (MDA) est indispensable à la **protection des données personnelles**, et donc au respect des dispositions imposées par la CNIL.

Enfin, la partie client de messagerie (MUA) doit également prendre en compte ces différents critères et répondre aux requêtes des différents serveurs.

La solution de confiance est complétée par l'utilisation d'un serveur de confiance faisant office d'Autorité de Certification et d'annuaire.

3. MOYENS DE MISE EN ŒUVRE

Afin de mettre en œuvre ces propositions de solutions, il convient de respecter un certain nombre d'étapes pour en assurer l'acceptation par la communauté scientifique, mais aussi par les utilisateurs.

1. Promotion de la signature électronique simple

La mise en œuvre de ces mécanismes de fiabilisation et de sécurisation de la messagerie électronique nécessite une phase de sensibilisation de l'usage généralisé de la signature électronique simple au sens de la Directive Européenne de 1999.

Il convient de mettre en avant la simplicité de délivrance de ces certificats, en gardant bien à l'esprit **les limites de confiance que cela implique au niveau de l'identité de l'émetteur tant que l'on utilise un certificat de Classe 1.**

Néanmoins, les avantages que cela procure, vis-à-vis de la fiabilité du contenu des messages (empreintes) et de la réduction intrinsèque de certaines catégories de « spams » doivent réussir à convaincre la communauté Internet du bien fondé de ces propositions.

2. Rédaction des RFC

La suite logique de cette thèse, et son objectif, commence par la rédaction formelle du RFC du protocole CEMTP décrit dans ce document, ainsi que des propositions de modifications des RFCs des protocoles POP3 et IMAP4, voire la proposition équivalente pour un nouveau protocole CEPOP.

Ces rédactions prendront du temps car elle devront se faire après la sensibilisation présentée ci-dessus et en regroupant, au sein de l'IETF, des membres qui adhèrent à ce projet.

Les bases de ces modifications sont intégrées dans cette thèse, et les parties impactées détaillées en Annexe N°1.

3. Développement d'une maquette

Comme dans la plupart des projets, il faut avancer et ne pas attendre.
Aussi, le meilleur moyen de promouvoir ces solutions est de développer une maquette dont les protocoles respecteront ces RFC et les contraintes complémentaires décrites dans cette thèse.

L'adhésion des utilisateurs qui verront leur taux de messages indésirables baisser sensiblement, et qui sauront immédiatement qu'un message fiable et clairement identifié vient d'un contact qu'ils connaissent, sera le meilleur moyen de convaincre la communauté scientifique de l'Internet.

Le développement d'un client de messagerie d'utilisation simple pouvant gérer l'ensemble de ces contraintes et la compatibilité entre les certificats

X509 et PGP sera un atout supplémentaire pour emporter l'acceptation de ces protocoles.

A ce titre, un rapprochement de plusieurs projets pourrait permettre une accélération sensible de la mise en œuvre d'une solution globale, et des projets comme « trustedbird » ou « openwebpki » seraient parfaitement adaptés.

4. UN SEUL OBJECTIF : LA CONFIANCE

Le besoin fondamental qui ressort des échanges avec des utilisateurs et des opérateurs depuis dix ans, est un accroissement sensible du niveau de confiance dans les échanges dématérialisés.

Dès qu'un objet ou un modèle devient virtuel, le niveau de confiance diminue, et l'on doit ajouter des mécanismes techniques de sécurité, ainsi que des contraintes juridiques fortes pour compenser cette perte de confiance.

Cet objectif a été le mien depuis la mise en place de la Loi du 13 Mars 2000 ayant modifié le Code civil.

L'analyse des besoins des utilisateurs et celle des causes de cette perte de confiance dans la messagerie électronique m'ont permis de rédiger les préceptes de cette thèse dans un seul but : **développer la confiance.**

XII. Bibliographie

[1] IETF Internet Engineering Task Force : RFCs
www.ietf.org

[2] ISO : International Standard Organisation : Normes
www.iso.org

[3] RFC EDITOR : www.rfc-editor.org

[4] Secuserve : Rapport annuel des menaces emails 2008
www.secuserve.com/fr/pdf/presse/panorama-2008-secuserve.htm

[5] CLUSIF : Panorama cybercriminalité, année 2010
www.clusif.asso.fr/fr/production/ouvrages/pdf/CLUSIF-2011-Panorama-Cybercriminalite-annee-2010.pdf

[6] MAAWG : Email security waraness and usage report 2010
www.maawg.org/2010-maawg-email-security-awareness-and-usage-report
Trust in Email Begins with Authentication

[7] The Radicati Group : Email Statistics Report 2010

[8] Anti-spam.net : Encyclopédie du courrier électronique et du spam
Listes noires, listes blanches : efficacité et utilisation

[9] ENISA : Anti-spam measures survey 2009
www.enisa.europa.eu

[10] SYMANTEC : March 2011 Intelligence Report

[11] Verisign : Security and Trust : The backbone of doing business over the Internet

[12] Georgia Tech : Revealing Botnet Membership using DNSBL CounterIntelligence

[13] ASTARO : The hidden dangers of spam : how SMBs can confront security risks and restore productivity

[14] Charles COPIN (Menaces) : L'usurpation d'identité évolue fortement dans le monde
www.wmaker.net/menaces

[15] CR2PA : Livre blanc de l'archivage des mails
www.cr2pa.fr

[16] SonicWall 2006 : Daniel J. Langin : A guide to keeping e-mail legal

[17] Trustedibird : Projet de dévelopement d'un logiciel de messagerie sécurisé
www.trustedbird.org

[18] Directive Européenne du 13 Décembre 1999
eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=CELEX:31999L0093:fr:HTML

[19] Loi n° 2000-230 du 13 mars 2000 portant adaptation du droit de la preuve aux technologies de l'information et relative à la signature électronique
www.legifrance.gouv.fr/affichTexte.do?cidTexte=LEGITEXT000005629200

[20] Décret n°2001-272 du 30 mars 2001 pris pour l'application de l'article 1316-4 du code civil et relatif à la signature électronique
www.legifrance.gouv.fr/affichTexte.do?cidTexte=LEGITEXT000005630796

[21] Loi n° 2004-575 du 21 juin 2004 pour la confiance dans l'économie numérique
ww.legifrance.gouv.fr/affichTexte.do?cidTexte=JORFTEXT000000801164&dateTexte=

[22] La protection du cyber-conommateur dans la loi pour la confiance dans l'économie numérique
Revue de Droit de l'Immatériel (Lamy 2005)

[23] TGI Rochefort sur Mer : 28/02/2001

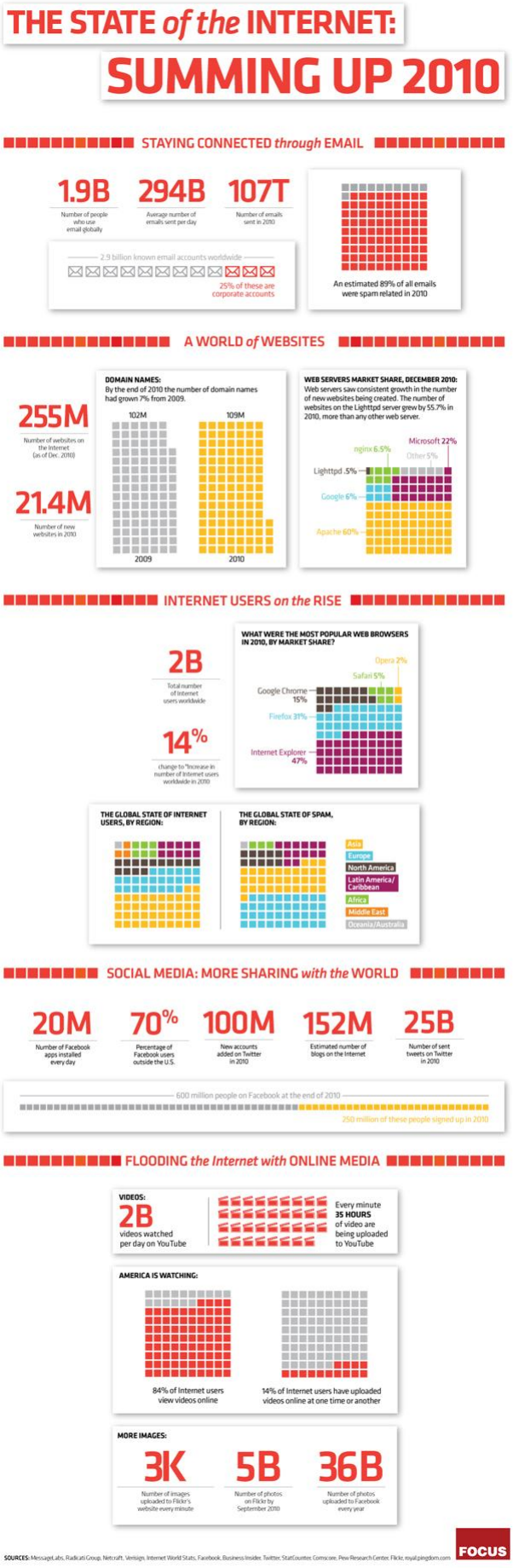
[24] TGI Paris : 15/01/2002

[25] Rapport CNIL : Le publipostage électronique et la protection des données personnelles (14/10/1999)

[26] FNTC : Fédération Nationale des Tiers de Confiance : Guide la signature électronique

www.fntc.org/component/option,com_remository/Itemid,19/func,startdown/id,231/

XIII. Table des annexes


THE STATE of the INTERNET:
SUMMING UP 2010
STAYING CONNECTED through EMAIL
1.9B
Number of people who use email globally
294B
Average number of emails sent per day
107T
Number of emails sent in 2010
2.9 billion known email accounts worldwide
25% of these are corporate accounts
An estimated 89% of all emails were spam related in 2010
A WORLD of WEBSITES
255M
Number of websites on the Internet (as of Dec. 2010)
21.4M
Number of new websites in 2010
DOMAIN NAMES:
By the end of 2010 the number of domain names had grown 7% from 2009.
102M
109M
2009
2010
WEB SERVERS MARKET SHARE, DECEMBER 2010:
Web servers saw consistent growth in the number of new websites being created. The number of websites on the Lighttpd server grew by 55.7% in 2010, more than any other web server.
nginx 6.5%
Microsoft 22%
Other 5%
Lighttpd .5%
Google 6%
Apache 60%
INTERNET USERS on the RISE
2B
Total number of Internet users worldwide
14%
change to "Increase in number of Internet users worldwide in 2010
WHAT WERE THE MOST POPULAR WEB BROWSERS IN 2010, BY MARKET SHARE?
Opera 2%
Safari 5%
Google Chrome 15%
Firefox 31%
Internet Explorer 47%
THE GLOBAL STATE OF INTERNET USERS, BY REGION:
THE GLOBAL STATE OF SPAM, BY REGION:
Asia
Europe
North America
Latin America/ Caribbean
Africa
Middle East
Oceania/Australia
SOCIAL MEDIA: MORE SHARING with the WORLD
20M
Number of Facebook apps installed every day
70%
Percentage of Facebook users outside the U.S.
100M
New accounts added on Twitter in 2010
152M
Estimated number of blogs on the Internet
25B
Number of sent tweets on Twitter in 2010
600 million people on Facebook at the end of 2010
250 million of these people signed up in 2010
FLOODING the Internet with ONLINE MEDIA
VIDEOS:
2B
videos watched per day on YouTube
Every minute 35 HOURS of video are being uploaded to YouTube
AMERICA IS WATCHING:
84% of Internet users view videos online
14% of Internet users have uploaded videos online at one time or another
MORE IMAGES:
3K
Number of images uploaded to Flickr's website every minute
5B
Number of photos on Flickr by September 2010
36B
Number of photos uploaded to Facebook every year
SOURCES: Messagelabs, Radicati Group, Netcraft, Verisign, Internet World Stats, Facebook, Business Insider, Twitter, StatCounter, Comscore, Pew Research Center, Flickr, royal.pingdom.com
FOCUS

Annexe 1

RFC N° 5321 SMTP (Simple Mail Transfer Protocol) en version intégrale dans lequel les passages impactés par les modifications demandées sont surlignés.

Network Working Group
Request for Comments: 5321
Obsoletes: 2821
Updates: 1123
Category: Standards Track

J. Klensin
October 2008

Simple Mail Transfer Protocol

Status of This Memo

This document specifies an Internet standards track protocol for the Internet community, and requests discussion and suggestions for improvements. Please refer to the current edition of the "Internet Official Protocol Standards" (STD 1) for the standardization state and status of this protocol. Distribution of this memo is unlimited.

Abstract

This document is a specification of the basic protocol for Internet electronic mail transport. It consolidates, updates, and clarifies several previous documents, making all or parts of most of them obsolete. It covers the SMTP extension mechanisms and best practices for the contemporary Internet, but does not provide details about particular extensions. Although SMTP was designed as a mail transport and delivery protocol, this specification also contains information that is important to its use as a "mail submission" protocol for "split-UA" (User Agent) mail reading systems and mobile environments.

Table of Contents

1. Introduction

1.1. Transport of Electronic Mail

The objective of the Simple Mail Transfer Protocol (SMTP) is to transfer mail reliably and efficiently.

SMTP is independent of the particular transmission subsystem and requires only a reliable ordered data stream channel. While this document specifically discusses transport over TCP, other transports are possible. Appendices to RFC 821 [1] describe some of them.

An important feature of SMTP is its capability to transport mail across multiple networks, usually referred to as "SMTP mail relaying" (see Section 3.6). A network consists of the mutually-TCP-accessible hosts on the public Internet, the mutually-TCP-accessible hosts on a firewall-isolated TCP/IP Intranet, or hosts in some other LAN or WAN environment utilizing a non-TCP transport-level protocol. Using SMTP, a process can transfer mail to another process on the same network or to some other network via a relay or gateway process accessible to both networks.

In this way, a mail message may pass through a number of intermediate relay or gateway hosts on its path from sender to ultimate recipient. The Mail eXchanger mechanisms of the domain name system (RFC 1035

[2], RFC 974 [12], and Section 5 of this document) are used to identify the appropriate next-hop destination for a message being transported.

1.2. History and Context for This Document

This document is a specification of the basic protocol for the Internet electronic mail transport. It consolidates, updates and clarifies, but does not add new or change existing functionality of the following:

- the original SMTP (Simple Mail Transfer Protocol) specification of RFC 821 [1],
- domain name system requirements and implications for mail transport from RFC 1035 [2] and RFC 974 [12],
- the clarifications and applicability statements in RFC 1123 [3], and
- material drawn from the SMTP Extension mechanisms in RFC 1869 [13].
- Editorial and clarification changes to RFC 2821 [14] to bring that specification to Draft Standard.

It obsoletes RFC 821, RFC 974, RFC 1869, and RFC 2821 and updates RFC 1123 (replacing the mail transport materials of RFC 1123). However, RFC 821 specifies some features that were not in significant use in the Internet by the mid-1990s and (in appendices) some additional transport models. Those sections are omitted here in the interest of clarity and brevity; readers needing them should refer to RFC 821.

It also includes some additional material from RFC 1123 that required amplification. This material has been identified in multiple ways, mostly by tracking flaming on various lists and newsgroups and problems of unusual readings or interpretations that have appeared as the SMTP extensions have been deployed. Where this specification moves beyond consolidation and actually differs from earlier documents, it supersedes them technically as well as textually.

Although SMTP was designed as a mail transport and delivery protocol, this specification also contains information that is important to its use as a "mail submission" protocol, as recommended for Post Office Protocol (POP) (RFC 937 [15], RFC 1939 [16]) and IMAP (RFC 3501 [17]). In general, the separate mail submission protocol specified in RFC 4409 [18] is now preferred to direct use of SMTP; more discussion of that subject appears in that document.

Section 2.3 provides definitions of terms specific to this document. Except when the historical terminology is necessary for clarity, this document uses the current 'client' and 'server' terminology to identify the sending and receiving SMTP processes, respectively.

A companion document, RFC 5322 [4], discusses message header sections and bodies and specifies formats and structures for them.

1.3. Document Conventions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [5]. As each of these terms was intentionally and carefully chosen to improve the interoperability of email, each use of these terms is to be treated as a conformance requirement.

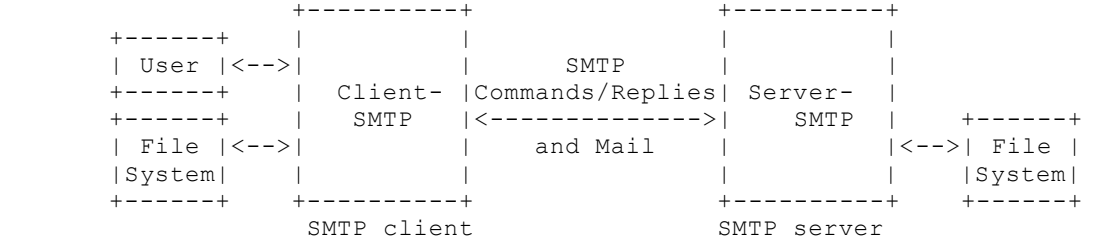
Because this document has a long history and to avoid the risk of various errors and of confusing readers and documents that point to

this one, most examples and the domain names they contain are preserved from RFC 2821. Readers are cautioned that these are illustrative examples that should not actually be used in either code or configuration files.

2. The SMTP Model

2.1. Basic Structure

The SMTP design can be pictured as:



When an SMTP client has a message to transmit, it establishes a two-way transmission channel to an SMTP server. The responsibility of an SMTP client is to transfer mail messages to one or more SMTP servers, or report its failure to do so.

The means by which a mail message is presented to an SMTP client, and how that client determines the identifier(s) ("names") of the domain(s) to which mail messages are to be transferred, is a local matter, and is not addressed by this document. In some cases, the designated domain(s), or those determined by an SMTP client, will identify the final destination(s) of the mail message. In other cases, common with SMTP clients associated with implementations of the POP (RFC 937 [15], RFC 1939 [16]) or IMAP (RFC 3501 [17]) protocols, or when the SMTP client is inside an isolated transport service environment, the domain determined will identify an intermediate destination through which all mail messages are to be relayed. SMTP clients that transfer all traffic regardless of the target domains associated with the individual messages, or that do not maintain queues for retrying message transmissions that initially cannot be completed, may otherwise conform to this specification but are not considered fully-capable. Fully-capable SMTP implementations, including the relays used by these less capable ones, and their destinations, are expected to support all of the queuing, retrying, and alternate address functions discussed in this specification. In many situations and configurations, the less-capable clients discussed above SHOULD be using the message submission protocol (RFC 4409 [18]) rather than SMTP.

The means by which an SMTP client, once it has determined a target domain, determines the identity of an SMTP server to which a copy of a message is to be transferred, and then performs that transfer, is covered by this document. To effect a mail transfer to an SMTP server, an SMTP client establishes a two-way transmission channel to that SMTP server. An SMTP client determines the address of an appropriate host running an SMTP server by resolving a destination domain name to either an intermediate Mail eXchanger host or a final target host.

An SMTP server may be either the ultimate destination or an intermediate "relay" (that is, it may assume the role of an SMTP client after receiving the message) or "gateway" (that is, it may transport the message further using some protocol other than SMTP). SMTP commands are generated by the SMTP client and sent to the SMTP server. SMTP replies are sent from the SMTP server to the SMTP client in response to the commands.

In other words, message transfer can occur in a single connection

between the original SMTP-sender and the final SMTP-recipient, or can occur in a series of hops through intermediary systems. In either case, once the server has issued a success response at the end of the mail data, a formal handoff of responsibility for the message occurs: the protocol requires that a server MUST accept responsibility for either delivering the message or properly reporting the failure to do so (see Sections 6.1, 6.2, and 7.8, below).

Once the transmission channel is established and initial handshaking is completed, the SMTP client normally initiates a mail transaction. Such a transaction consists of a series of commands to specify the originator and destination of the mail and transmission of the message content (including any lines in the header section or other structure) itself. When the same message is sent to multiple recipients, this protocol encourages the transmission of only one copy of the data for all recipients at the same destination (or intermediate relay) host.

The server responds to each command with a reply; replies may indicate that the command was accepted, that additional commands are expected, or that a temporary or permanent error condition exists. Commands specifying the sender or recipients may include server-permitted SMTP service extension requests, as discussed in Section 2.2. The dialog is purposely lock-step, one-at-a-time, although this can be modified by mutually agreed upon extension requests such as command pipelining (RFC 2920 [19]).

Once a given mail message has been transmitted, the client may either request that the connection be shut down or may initiate other mail transactions. In addition, an SMTP client may use a connection to an SMTP server for ancillary services such as verification of email addresses or retrieval of mailing list subscriber addresses.

As suggested above, this protocol provides mechanisms for the transmission of mail. Historically, this transmission normally occurred directly from the sending user's host to the receiving user's host when the two hosts are connected to the same transport service. When they are not connected to the same transport service, transmission occurs via one or more relay SMTP servers. A very common case in the Internet today involves submission of the original message to an intermediate, "message submission" server, which is similar to a relay but has some additional properties; such servers are discussed in Section 2.3.10 and at some length in RFC 4409 [18]. An intermediate host that acts as either an SMTP relay or as a gateway into some other transmission environment is usually selected through the use of the domain name service (DNS) Mail eXchanger mechanism.

Usually, intermediate hosts are determined via the DNS MX record, not by explicit "source" routing (see Section 5 and Appendix C and Appendix F.2).

2.2. The Extension Model

2.2.1. Background

In an effort that started in 1990, approximately a decade after RFC 821 was completed, the protocol was modified with a "service extensions" model that permits the client and server to agree to utilize shared functionality beyond the original SMTP requirements. The SMTP extension mechanism defines a means whereby an extended SMTP client and server may recognize each other, and the server can inform the client as to the service extensions that it supports.

Contemporary SMTP implementations MUST support the basic extension mechanisms. For instance, servers MUST support the EHLO command even if they do not implement any specific extensions and clients SHOULD preferentially utilize EHLO rather than HELO. (However, for

compatibility with older conforming implementations, SMTP clients and servers MUST support the original HELO mechanisms as a fallback.) Unless the different characteristics of HELO must be identified for interoperability purposes, this document discusses only EHLO.

SMTP is widely deployed and high-quality implementations have proven to be very robust. However, the Internet community now considers some services to be important that were not anticipated when the protocol was first designed. If support for those services is to be added, it must be done in a way that permits older implementations to continue working acceptably. The extension framework consists of:

- The SMTP command EHLO, superseding the earlier HELO,
- a registry of SMTP service extensions,
- additional parameters to the SMTP MAIL and RCPT commands, and
- optional replacements for commands defined in this protocol, such as for DATA in non-ASCII transmissions (RFC 3030 [20]).

SMTP's strength comes primarily from its simplicity. Experience with many protocols has shown that protocols with few options tend towards ubiquity, whereas protocols with many options tend towards obscurity.

Each and every extension, regardless of its benefits, must be carefully scrutinized with respect to its implementation, deployment, and interoperability costs. In many cases, the cost of extending the SMTP service will likely outweigh the benefit.

2.2.2. Definition and Registration of Extensions

The IANA maintains a registry of SMTP service extensions. A corresponding EHLO keyword value is associated with each extension. Each service extension registered with the IANA must be defined in a formal Standards-Track or IESG-approved Experimental protocol document. The definition must include:

- the textual name of the SMTP service extension;
- the EHLO keyword value associated with the extension;
- the syntax and possible values of parameters associated with the EHLO keyword value;
- any additional SMTP verbs associated with the extension (additional verbs will usually be, but are not required to be, the same as the EHLO keyword value);
- any new parameters the extension associates with the MAIL or RCPT verbs;
- a description of how support for the extension affects the behavior of a server and client SMTP; and
- the increment by which the extension is increasing the maximum length of the commands MAIL and/or RCPT, over that specified in this Standard.

In addition, any EHLO keyword value starting with an upper or lower case "X" refers to a local SMTP service extension used exclusively through bilateral agreement. Keywords beginning with "X" MUST NOT be used in a registered service extension. Conversely, keyword values presented in the EHLO response that do not begin with "X" MUST correspond to a Standard, Standards-Track, or IESG-approved Experimental SMTP service extension registered with IANA. A conforming server MUST NOT offer non-"X"-prefixed keyword values that are not described in a registered extension.

Additional verbs and parameter names are bound by the same rules as EHLO keywords; specifically, verbs beginning with "X" are local extensions that may not be registered or standardized. Conversely, verbs not beginning with "X" must always be registered.

2.2.3. Special Issues with Extensions

Extensions that change fairly basic properties of SMTP operation are permitted. The text in other sections of this document must be understood in that context. In particular, extensions can change the minimum limits specified in Section 4.5.3, can change the ASCII character set requirement as mentioned above, or can introduce some optional modes of message handling.

In particular, if an extension implies that the delivery path normally supports special features of that extension, and an intermediate SMTP system finds a next hop that does not support the required extension, it MAY choose, based on the specific extension and circumstances, to requeue the message and try later and/or try an alternate MX host. If this strategy is employed, the timeout to fall back to an unextended format (if one is available) SHOULD be less than the normal timeout for bouncing as undeliverable (e.g., if normal timeout is three days, the requeue timeout before attempting to transmit the mail without the extension might be one day).

2.3. SMTP Terminology

2.3.1. Mail Objects

SMTP transports a mail object. A mail object contains an envelope and content.

The SMTP envelope is sent as a series of SMTP protocol units (described in Section 3). It consists of an originator address (to which error reports should be directed), one or more recipient addresses, and optional protocol extension material. Historically, variations on the reverse-path (originator) address specification command (MAIL) could be used to specify alternate delivery modes, such as immediate display; those variations have now been deprecated (see Appendix F and Appendix F.6).

The SMTP content is sent in the SMTP DATA protocol unit and has two parts: the header section and the body. If the content conforms to other contemporary standards, the header section consists of a collection of header fields, each consisting of a header name, a colon, and data, structured as in the message format specification (RFC 5322 [4]); the body, if structured, is defined according to MIME (RFC 2045 [21]). The content is textual in nature, expressed using the US-ASCII repertoire [6]. Although SMTP extensions (such as "8BITMIME", RFC 1652 [22]) may relax this restriction for the content body, the content header fields are always encoded using the US-ASCII repertoire. Two MIME extensions (RFC 2047 [23] and RFC 2231 [24]) define an algorithm for representing header values outside the US-ASCII repertoire, while still encoding them using the US-ASCII repertoire.

2.3.2. Senders and Receivers

In RFC 821, the two hosts participating in an SMTP transaction were described as the "SMTP-sender" and "SMTP-receiver". This document has been changed to reflect current industry terminology and hence refers to them as the "SMTP client" (or sometimes just "the client") and "SMTP server" (or just "the server"), respectively. Since a given host may act both as server and client in a relay situation, "receiver" and "sender" terminology is still used where needed for clarity.

2.3.3. Mail Agents and Message Stores

Additional mail system terminology became common after RFC 821 was published and, where convenient, is used in this specification. In particular, SMTP servers and clients provide a mail transport service and therefore act as "Mail Transfer Agents" (MTAs). "Mail User Agents" (MUAs or UAs) are normally thought of as the sources and targets of mail. At the source, an MUA might collect mail to be transmitted from a user and hand it off to an MTA; the final ("delivery") MTA would be thought of as handing the mail off to an MUA (or at least transferring responsibility to it, e.g., by depositing the message in a "message store"). However, while these terms are used with at least the appearance of great precision in other environments, the implied boundaries between MUAs and MTAs often do not accurately match common, and conforming, practices with Internet mail. Hence, the reader should be cautious about inferring the strong relationships and responsibilities that might be implied if these terms were used elsewhere.

2.3.4. Host

For the purposes of this specification, a host is a computer system attached to the Internet (or, in some cases, to a private TCP/IP network) and supporting the SMTP protocol. Hosts are known by names (see the next section); they SHOULD NOT be identified by numerical addresses, i.e., by address literals as described in Section 4.1.2.

2.3.5. Domain Names

A domain name (or often just a "domain") consists of one or more components, separated by dots if more than one appears. In the case of a top-level domain used by itself in an email address, a single string is used without any dots. This makes the requirement, described in more detail below, that only fully-qualified domain names appear in SMTP transactions on the public Internet, particularly important where top-level domains are involved. These components ("labels" in DNS terminology, RFC 1035 [2]) are restricted for SMTP purposes to consist of a sequence of letters, digits, and hyphens drawn from the ASCII character set [6]. Domain names are used as names of hosts and of other entities in the domain name hierarchy. For example, a domain may refer to an alias (label of a CNAME RR) or the label of Mail eXchanger records to be used to deliver mail instead of representing a host name. See RFC 1035 [2] and Section 5 of this specification.

The domain name, as described in this document and in RFC 1035 [2], is the entire, fully-qualified name (often referred to as an "FQDN"). A domain name that is not in FQDN form is no more than a local alias. Local aliases MUST NOT appear in any SMTP transaction.

Only resolvable, fully-qualified domain names (FQDNs) are permitted when domain names are used in SMTP. In other words, names that can be resolved to MX RRs or address (i.e., A or AAAA) RRs (as discussed in Section 5) are permitted, as are CNAME RRs whose targets can be resolved, in turn, to MX or address RRs. Local nicknames or unqualified names MUST NOT be used. There are two exceptions to the rule requiring FQDNs:

- The domain name given in the EHLO command MUST be either a primary host name (a domain name that resolves to an address RR) or, if the host has no name, an address literal, as described in Section 4.1.3 and discussed further in the EHLO discussion of Section 4.1.4.

- The reserved mailbox name "postmaster" may be used in a RCPT command without domain qualification (see Section 4.1.1.3) and MUST be accepted if so used.

2.3.6. Buffer and State Table

SMTP sessions are stateful, with both parties carefully maintaining a common view of the current state. In this document, we model this state by a virtual "buffer" and a "state table" on the server that may be used by the client to, for example, "clear the buffer" or "reset the state table", causing the information in the buffer to be discarded and the state to be returned to some previous state.

2.3.7. Commands and Replies

SMTP commands and, unless altered by a service extension, message data, are transmitted from the sender to the receiver via the transmission channel in "lines".

An SMTP reply is an acknowledgment (positive or negative) sent in "lines" from receiver to sender via the transmission channel in response to a command. The general form of a reply is a numeric completion code (indicating failure or success) usually followed by a text string. The codes are for use by programs and the text is usually intended for human users. RFC 3463 [25], specifies further structuring of the reply strings, including the use of supplemental and more specific completion codes (see also RFC 5248 [26]).

2.3.8. Lines

Lines consist of zero or more data characters terminated by the sequence ASCII character "CR" (hex value 0D) followed immediately by ASCII character "LF" (hex value 0A). This termination sequence is denoted as <CRLF> in this document. Conforming implementations MUST NOT recognize or generate any other character or character sequence as a line terminator. Limits MAY be imposed on line lengths by servers (see Section 4).

In addition, the appearance of "bare" "CR" or "LF" characters in text (i.e., either without the other) has a long history of causing problems in mail implementations and applications that use the mail system as a tool. SMTP client implementations MUST NOT transmit these characters except when they are intended as line terminators and then MUST, as indicated above, transmit them only as a <CRLF> sequence.

2.3.9. Message Content and Mail Data

The terms "message content" and "mail data" are used interchangeably in this document to describe the material transmitted after the DATA command is accepted and before the end of data indication is transmitted. Message content includes the message header section and the possibly structured message body. The MIME specification (RFC 2045 [21]) provides the standard mechanisms for structured message bodies.

2.3.10. Originator, Delivery, Relay, and Gateway Systems

This specification makes a distinction among four types of SMTP systems, based on the role those systems play in transmitting electronic mail. An "originating" system (sometimes called an SMTP originator) introduces mail into the Internet or, more generally, into a transport service environment. A "delivery" SMTP system is one that receives mail from a transport service environment and passes it to a mail user agent or deposits it in a message store that a mail user agent is expected to subsequently access. A "relay" SMTP system (usually referred to just as a "relay") receives mail from an SMTP client and transmits it, without modification to the message data other than adding trace information, to another SMTP server for further relaying or for delivery.

A "gateway" SMTP system (usually referred to just as a "gateway")

receives mail from a client system in one transport environment and transmits it to a server system in another transport environment. Differences in protocols or message semantics between the transport environments on either side of a gateway may require that the gateway system perform transformations to the message that are not permitted to SMTP relay systems. For the purposes of this specification, firewalls that rewrite addresses should be considered as gateways, even if SMTP is used on both sides of them (see RFC 2979 [27]).

2.3.11. Mailbox and Address

As used in this specification, an "address" is a character string that identifies a user to whom mail will be sent or a location into which mail will be deposited. The term "mailbox" refers to that depository. The two terms are typically used interchangeably unless the distinction between the location in which mail is placed (the mailbox) and a reference to it (the address) is important. An address normally consists of user and domain specifications. The standard mailbox naming convention is defined to be "local-part@domain"; contemporary usage permits a much broader set of applications than simple "user names". Consequently, and due to a long history of problems when intermediate hosts have attempted to optimize transport by modifying them, the local-part MUST be interpreted and assigned semantics only by the host specified in the domain part of the address.

2.4. General Syntax Principles and Transaction Model

SMTP commands and replies have a rigid syntax. All commands begin with a command verb. All replies begin with a three digit numeric code. In some commands and replies, arguments are required following the verb or reply code. Some commands do not accept arguments (after the verb), and some reply codes are followed, sometimes optionally, by free form text. In both cases, where text appears, it is separated from the verb or reply code by a space character. Complete definitions of commands and replies appear in Section 4.

Verbs and argument values (e.g., "TO:" or "to:" in the RCPT command and extension name keywords) are not case sensitive, with the sole exception in this specification of a mailbox local-part (SMTP Extensions may explicitly specify case-sensitive elements). That is, a command verb, an argument value other than a mailbox local-part, and free form text MAY be encoded in upper case, lower case, or any mixture of upper and lower case with no impact on its meaning. The local-part of a mailbox MUST BE treated as case sensitive. Therefore, SMTP implementations MUST take care to preserve the case of mailbox local-parts. In particular, for some hosts, the user "smith" is different from the user "Smith". However, exploiting the case sensitivity of mailbox local-parts impedes interoperability and is discouraged. Mailbox domains follow normal DNS rules and are hence not case sensitive.

A few SMTP servers, in violation of this specification (and RFC 821) require that command verbs be encoded by clients in upper case. Implementations MAY wish to employ this encoding to accommodate those servers.

The argument clause consists of a variable-length character string ending with the end of the line, i.e., with the character sequence <CRLF>. The receiver will take no action until this sequence is received.

The syntax for each command is shown with the discussion of that command. Common elements and parameters are shown in Section 4.1.2.

Commands and replies are composed of characters from the ASCII character set [6]. When the transport service provides an 8-bit byte (octet) transmission channel, each 7-bit character is transmitted,

right justified, in an octet with the high-order bit cleared to zero. More specifically, the unextended SMTP service provides 7-bit transport only. An originating SMTP client that has not successfully negotiated an appropriate extension with a particular server (see the next paragraph) MUST NOT transmit messages with information in the high-order bit of octets. If such messages are transmitted in violation of this rule, receiving SMTP servers MAY clear the high-order bit or reject the message as invalid. In general, a relay SMTP SHOULD assume that the message content it has received is valid and, assuming that the envelope permits doing so, relay it without inspecting that content. Of course, if the content is mislabeled and the data path cannot accept the actual content, this may result in the ultimate delivery of a severely garbled message to the recipient. Delivery SMTP systems MAY reject such messages, or return them as undeliverable, rather than deliver them. In the absence of a server-offered extension explicitly permitting it, a sending SMTP system is not permitted to send envelope commands in any character set other than US-ASCII. Receiving systems SHOULD reject such commands, normally using "500 syntax error - invalid character" replies.

8-bit message content transmission MAY be requested of the server by a client using extended SMTP facilities, notably the "8BITMIME" extension, RFC 1652 [22]. 8BITMIME SHOULD be supported by SMTP servers. However, it MUST NOT be construed as authorization to transmit unrestricted 8-bit material, nor does 8BITMIME authorize transmission of any envelope material in other than ASCII. 8BITMIME MUST NOT be requested by senders for material with the high bit on that is not in MIME format with an appropriate content-transfer encoding; servers MAY reject such messages.

The metalinguistic notation used in this document corresponds to the "Augmented BNF" used in other Internet mail system documents. The reader who is not familiar with that syntax should consult the ABNF specification in RFC 5234 [7]. Metalanguage terms used in running text are surrounded by pointed brackets (e.g., <CRLF>) for clarity. The reader is cautioned that the grammar expressed in the metalanguage is not comprehensive. There are many instances in which provisions in the text constrain or otherwise modify the syntax or semantics implied by the grammar.

3. The SMTP Procedures: An Overview

This section contains descriptions of the procedures used in SMTP: session initiation, mail transaction, forwarding mail, verifying mailbox names and expanding mailing lists, and opening and closing exchanges. Comments on relaying, a note on mail domains, and a discussion of changing roles are included at the end of this section. Several complete scenarios are presented in Appendix D.

3.1. Session Initiation

An SMTP session is initiated when a client opens a connection to a server and the server responds with an opening message.

SMTP server implementations MAY include identification of their software and version information in the connection greeting reply after the 220 code, a practice that permits more efficient isolation and repair of any problems. Implementations MAY make provision for SMTP servers to disable the software and version announcement where it causes security concerns. While some systems also identify their contact point for mail problems, this is not a substitute for maintaining the required "postmaster" address (see Section 4).

The SMTP protocol allows a server to formally reject a mail session while still allowing the initial connection as follows: a 554 response MAY be given in the initial connection opening message instead of the 220. A server taking this approach MUST still wait for the client to send a QUIT (see Section 4.1.1.10) before closing

the connection and SHOULD respond to any intervening commands with "503 bad sequence of commands". Since an attempt to make an SMTP connection to such a system is probably in error, a server returning a 554 response on connection opening SHOULD provide enough information in the reply text to facilitate debugging of the sending system.

3.2. Client Initiation

Once the server has sent the greeting (welcoming) message and the client has received it, the client normally sends the EHLO command to the server, indicating the client's identity. In addition to opening the session, use of EHLO indicates that the client is able to process service extensions and requests that the server provide a list of the extensions it supports. Older SMTP systems that are unable to support service extensions, and contemporary clients that do not require service extensions in the mail session being initiated, MAY use HELO instead of EHLO. Servers MUST NOT return the extended EHLO-style response to a HELO command. For a particular connection attempt, if the server returns a "command not recognized" response to EHLO, the client SHOULD be able to fall back and send HELO.

In the EHLO command, the host sending the command identifies itself; the command may be interpreted as saying "Hello, I am <domain>" (and, in the case of EHLO, "and I support service extension requests").

3.3. Mail Transactions

There are three steps to SMTP mail transactions. The transaction starts with a MAIL command that gives the sender identification. (In general, the MAIL command may be sent only when no mail transaction is in progress; see Section 4.1.4.) A series of one or more RCPT commands follows, giving the receiver information. Then, a DATA command initiates transfer of the mail data and is terminated by the "end of mail" data indicator, which also confirms the transaction.

The first step in the procedure is the MAIL command.

MAIL FROM:<reverse-path> [SP <mail-parameters>] <CRLF>

This command tells the SMTP-receiver that a new mail transaction is starting and to reset all its state tables and buffers, including any recipients or mail data. The <reverse-path> portion of the first or only argument contains the source mailbox (between "<" and ">" brackets), which can be used to report errors (see Section 4.2 for a discussion of error reporting). If accepted, the SMTP server returns a "250 OK" reply. If the mailbox specification is not acceptable for some reason, the server MUST return a reply indicating whether the failure is permanent (i.e., will occur again if the client tries to send the same address again) or temporary (i.e., the address might be accepted if the client tries again later). Despite the apparent scope of this requirement, there are circumstances in which the acceptability of the reverse-path may not be determined until one or more forward-paths (in RCPT commands) can be examined. In those cases, the server MAY reasonably accept the reverse-path (with a 250 reply) and then report problems after the forward-paths are received and examined. Normally, failures produce 550 or 553 replies.

Historically, the <reverse-path> was permitted to contain more than just a mailbox; however, contemporary systems SHOULD NOT use source routing (see Appendix C).

The optional <mail-parameters> are associated with negotiated SMTP service extensions (see Section 2.2).

The second step in the procedure is the RCPT command. This step of the procedure can be repeated any number of times.

```
    RCPT TO:<forward-path> [ SP <rcpt-parameters> ] <CRLF>
```

The first or only argument to this command includes a forward-path (normally a mailbox and domain, always surrounded by "<" and ">" brackets) identifying one recipient. If accepted, the SMTP server returns a "250 OK" reply and stores the forward-path. If the recipient is known not to be a deliverable address, the SMTP server returns a 550 reply, typically with a string such as "no such user - " and the mailbox name (other circumstances and reply codes are possible).

The <forward-path> can contain more than just a mailbox. Historically, the <forward-path> was permitted to contain a source routing list of hosts and the destination mailbox; however, contemporary SMTP clients SHOULD NOT utilize source routes (see Appendix C). Servers MUST be prepared to encounter a list of source routes in the forward-path, but they SHOULD ignore the routes or MAY decline to support the relaying they imply. Similarly, servers MAY decline to accept mail that is destined for other hosts or systems. These restrictions make a server useless as a relay for clients that do not support full SMTP functionality. Consequently, restricted-capability clients MUST NOT assume that any SMTP server on the Internet can be used as their mail processing (relaying) site. If a RCPT command appears without a previous MAIL command, the server MUST return a 503 "Bad sequence of commands" response. The optional <rcpt-parameters> are associated with negotiated SMTP service extensions (see Section 2.2).

Since it has been a common source of errors, it is worth noting that spaces are not permitted on either side of the colon following FROM in the MAIL command or TO in the RCPT command. The syntax is exactly as given above.

The third step in the procedure is the DATA command (or some alternative specified in a service extension).

```
    DATA <CRLF>
```

If accepted, the SMTP server returns a 354 Intermediate reply and considers all succeeding lines up to but not including the end of mail data indicator to be the message text. When the end of text is successfully received and stored, the SMTP-receiver sends a "250 OK" reply.

Since the mail data is sent on the transmission channel, the end of mail data must be indicated so that the command and reply dialog can be resumed. SMTP indicates the end of the mail data by sending a line containing only a "." (period or full stop). A transparency procedure is used to prevent this from interfering with the user's text (see Section 4.5.2).

The end of mail data indicator also confirms the mail transaction and tells the SMTP server to now process the stored recipients and mail data. If accepted, the SMTP server returns a "250 OK" reply. The DATA command can fail at only two points in the protocol exchange:

If there was no MAIL, or no RCPT, command, or all such commands were rejected, the server MAY return a "command out of sequence" (503) or "no valid recipients" (554) reply in response to the DATA command. If one of those replies (or any other 5yz reply) is received, the client MUST NOT send the message data; more generally, message data MUST NOT be sent unless a 354 reply is received.

If the verb is initially accepted and the 354 reply issued, the DATA command should fail only if the mail transaction was incomplete (for example, no recipients), if resources were unavailable (including, of course, the server unexpectedly becoming unavailable), or if the server determines that the message should be rejected for policy or

other reasons.

However, in practice, some servers do not perform recipient verification until after the message text is received. These servers SHOULD treat a failure for one or more recipients as a "subsequent failure" and return a mail message as discussed in Section 6 and, in particular, in Section 6.1. Using a "550 mailbox not found" (or equivalent) reply code after the data are accepted makes it difficult or impossible for the client to determine which recipients failed.

When the RFC 822 format ([28], [4]) is being used, the mail data include the header fields such as those named Date, Subject, To, Cc, and From. Server SMTP systems SHOULD NOT reject messages based on perceived defects in the RFC 822 or MIME (RFC 2045 [21]) message header section or message body. In particular, they MUST NOT reject messages in which the numbers of Resent-header fields do not match or Resent-to appears without Resent-from and/or Resent-date.

Mail transaction commands MUST be used in the order discussed above.

3.4. Forwarding for Address Correction or Updating

Forwarding support is most often required to consolidate and simplify addresses within, or relative to, some enterprise and less frequently to establish addresses to link a person's prior address with a current one. Silent forwarding of messages (without server notification to the sender), for security or non-disclosure purposes, is common in the contemporary Internet.

In both the enterprise and the "new address" cases, information hiding (and sometimes security) considerations argue against exposure of the "final" address through the SMTP protocol as a side effect of the forwarding activity. This may be especially important when the final address may not even be reachable by the sender. Consequently, the "forwarding" mechanisms described in Section 3.2 of RFC 821, and especially the 251 (corrected destination) and 551 reply codes from RCPT must be evaluated carefully by implementers and, when they are available, by those configuring systems (see also Section 7.4).

In particular:

- Servers MAY forward messages when they are aware of an address change. When they do so, they MAY either provide address-updating information with a 251 code, or may forward "silently" and return a 250 code. However, if a 251 code is used, they MUST NOT assume that the client will actually update address information or even return that information to the user.

Alternately,

- Servers MAY reject messages or return them as non-deliverable when they cannot be delivered precisely as addressed. When they do so, they MAY either provide address-updating information with a 551 code, or may reject the message as undeliverable with a 550 code and no address-specific information. However, if a 551 code is used, they MUST NOT assume that the client will actually update address information or even return that information to the user.

SMTP server implementations that support the 251 and/or 551 reply codes SHOULD provide configuration mechanisms so that sites that conclude that they would undesirably disclose information can disable or restrict their use.

3.5. Commands for Debugging Addresses

3.5.1. Overview

SMTP provides commands to verify a user name or obtain the content of

a mailing list. This is done with the VRFY and EXPN commands, which have character string arguments. Implementations SHOULD support VRFY and EXPN (however, see Section 3.5.2 and Section 7.3).

For the VRFY command, the string is a user name or a user name and domain (see below). If a normal (i.e., 250) response is returned, the response MAY include the full name of the user and MUST include the mailbox of the user. It MUST be in either of the following forms:

```
User Name <local-part@domain>
local-part@domain
```

When a name that is the argument to VRFY could identify more than one mailbox, the server MAY either note the ambiguity or identify the alternatives. In other words, any of the following are legitimate responses to VRFY:

```
553 User ambiguous
```

or

```
553- Ambiguous; Possibilities are
553-Joe Smith <jsmith@foo.com>
553-Harry Smith <hsmith@foo.com>
553 Melvin Smith <dweep@foo.com>
```

or

```
553-Ambiguous; Possibilities
553- <jsmith@foo.com>
553- <hsmith@foo.com>
553 <dweep@foo.com>
```

Under normal circumstances, a client receiving a 553 reply would be expected to expose the result to the user. Use of exactly the forms given, and the "user ambiguous" or "ambiguous" keywords, possibly supplemented by extended reply codes, such as those described in RFC 3463 [25], will facilitate automated translation into other languages as needed. Of course, a client that was highly automated or that was operating in another language than English might choose to try to translate the response to return some other indication to the user than the literal text of the reply, or to take some automated action such as consulting a directory service for additional information before reporting to the user.

For the EXPN command, the string identifies a mailing list, and the successful (i.e., 250) multiline response MAY include the full name of the users and MUST give the mailboxes on the mailing list.

In some hosts, the distinction between a mailing list and an alias for a single mailbox is a bit fuzzy, since a common data structure may hold both types of entries, and it is possible to have mailing lists containing only one mailbox. If a request is made to apply VRFY to a mailing list, a positive response MAY be given if a message so addressed would be delivered to everyone on the list, otherwise an error SHOULD be reported (e.g., "550 That is a mailing list, not a user" or "252 Unable to verify members of mailing list"). If a request is made to expand a user name, the server MAY return a positive response consisting of a list containing one name, or an error MAY be reported (e.g., "550 That is a user name, not a mailing list").

In the case of a successful multiline reply (normal for EXPN), exactly one mailbox is to be specified on each line of the reply. The case of an ambiguous request is discussed above.

"User name" is a fuzzy term and has been used deliberately. An

implementation of the VRFY or EXPN commands MUST include at least recognition of local mailboxes as "user names". However, since current Internet practice often results in a single host handling mail for multiple domains, hosts, especially hosts that provide this functionality, SHOULD accept the "local-part@domain" form as a "user name"; hosts MAY also choose to recognize other strings as "user names".

The case of expanding a mailbox list requires a multiline reply, such as:

```
C: EXPN Example-People
S: 250-Jon Postel <Postel@isi.edu>
S: 250-Fred Fonebone <Fonebone@physics.foo-u.edu>
S: 250 Sam Q. Smith <SQSmith@specific.generic.com>
```

or

```
C: EXPN Executive-Washroom-List
S: 550 Access Denied to You.
```

The character string arguments of the VRFY and EXPN commands cannot be further restricted due to the variety of implementations of the user name and mailbox list concepts. On some systems, it may be appropriate for the argument of the EXPN command to be a file name for a file containing a mailing list, but again there are a variety of file naming conventions in the Internet. Similarly, historical variations in what is returned by these commands are such that the response SHOULD be interpreted very carefully, if at all, and SHOULD generally only be used for diagnostic purposes.

3.5.2. VRFY Normal Response

When normal (2yz or 551) responses are returned from a VRFY or EXPN request, the reply MUST include the <Mailbox> name using a "<local-part@domain>" construction, where "domain" is a fully-qualified domain name. In circumstances exceptional enough to justify violating the intent of this specification, free-form text MAY be returned. In order to facilitate parsing by both computers and people, addresses SHOULD appear in pointed brackets. When addresses, rather than free-form debugging information, are returned, EXPN and VRFY MUST return only valid domain addresses that are usable in SMTP RCPT commands. Consequently, if an address implies delivery to a program or other system, the mailbox name used to reach that target MUST be given. Paths (explicit source routes) MUST NOT be returned by VRFY or EXPN.

Server implementations SHOULD support both VRFY and EXPN. For security reasons, implementations MAY provide local installations a way to disable either or both of these commands through configuration options or the equivalent (see Section 7.3). When these commands are supported, they are not required to work across relays when relaying is supported. Since they were both optional in RFC 821, but VRFY was made mandatory in RFC 1123 [3], if EXPN is supported, it MUST be listed as a service extension in an EHLO response. VRFY MAY be listed as a convenience but, since support for it is required, SMTP clients are not required to check for its presence on the extension list before using it.

3.5.3. Meaning of VRFY or EXPN Success Response

A server MUST NOT return a 250 code in response to a VRFY or EXPN command unless it has actually verified the address. In particular, a server MUST NOT return 250 if all it has done is to verify that the syntax given is valid. In that case, 502 (Command not implemented) or 500 (Syntax error, command unrecognized) SHOULD be returned. As stated elsewhere, implementation (in the sense of actually validating addresses and returning information) of VRFY and EXPN are strongly

recommended. Hence, implementations that return 500 or 502 for VRFY are not in full compliance with this specification.

There may be circumstances where an address appears to be valid but cannot reasonably be verified in real time, particularly when a server is acting as a mail exchanger for another server or domain. "Apparent validity", in this case, would normally involve at least syntax checking and might involve verification that any domains specified were ones to which the host expected to be able to relay mail. In these situations, reply code 252 SHOULD be returned. These cases parallel the discussion of RCPT verification in Section 2.1. Similarly, the discussion in Section 3.4 applies to the use of reply codes 251 and 551 with VRFY (and EXPN) to indicate addresses that are recognized but that would be forwarded or rejected were mail received for them. Implementations generally SHOULD be more aggressive about address verification in the case of VRFY than in the case of RCPT, even if it takes a little longer to do so.

3.5.4. Semantics and Applications of EXPN

EXPN is often very useful in debugging and understanding problems with mailing lists and multiple-target-address aliases. Some systems have attempted to use source expansion of mailing lists as a means of eliminating duplicates. The propagation of aliasing systems with mail on the Internet for hosts (typically with MX and CNAME DNS records), for mailboxes (various types of local host aliases), and in various proxying arrangements has made it nearly impossible for these strategies to work consistently, and mail systems SHOULD NOT attempt them.

3.6. Relaying and Mail Routing

3.6.1. Source Routes and Relaying

In general, the availability of Mail eXchanger records in the domain name system (RFC 1035 [2], RFC 974 [12]) makes the use of explicit source routes in the Internet mail system unnecessary. Many historical problems with the interpretation of explicit source routes have made their use undesirable. SMTP clients SHOULD NOT generate explicit source routes except under unusual circumstances. SMTP servers MAY decline to act as mail relays or to accept addresses that specify source routes. When route information is encountered, SMTP servers MAY ignore the route information and simply send to the final destination specified as the last element in the route and SHOULD do so. There has been an invalid practice of using names that do not appear in the DNS as destination names, with the senders counting on the intermediate hosts specified in source routing to resolve any problems. If source routes are stripped, this practice will cause failures. This is one of several reasons why SMTP clients MUST NOT generate invalid source routes or depend on serial resolution of names.

When source routes are not used, the process described in RFC 821 for constructing a reverse-path from the forward-path is not applicable and the reverse-path at the time of delivery will simply be the address that appeared in the MAIL command.

3.6.2. Mail eXchange Records and Relaying

A relay SMTP server is usually the target of a DNS MX record that designates it, rather than the final delivery system. The relay server may accept or reject the task of relaying the mail in the same way it accepts or rejects mail for a local user. If it accepts the task, it then becomes an SMTP client, establishes a transmission channel to the next SMTP server specified in the DNS (according to the rules in Section 5), and sends it the mail. If it declines to relay mail to a particular address for policy reasons, a 550 response SHOULD be returned.

This specification does not deal with the verification of return paths for use in delivery notifications. Recent work, such as that on SPF [29] and DKIM [30] [31], has been done to provide ways to ascertain that an address is valid or belongs to the person who actually sent the message. A server MAY attempt to verify the return path before using its address for delivery notifications, but methods of doing so are not defined here nor is any particular method recommended at this time.

3.6.3. Message Submission Servers as Relays

Many mail-sending clients exist, especially in conjunction with facilities that receive mail via POP3 or IMAP, that have limited capability to support some of the requirements of this specification, such as the ability to queue messages for subsequent delivery attempts. For these clients, it is common practice to make private arrangements to send all messages to a single server for processing and subsequent distribution. SMTP, as specified here, is not ideally suited for this role. A standardized mail submission protocol has been developed that is gradually superseding practices based on SMTP (see RFC 4409 [18]). In any event, because these arrangements are private and fall outside the scope of this specification, they are not described here.

It is important to note that MX records can point to SMTP servers that act as gateways into other environments, not just SMTP relays and final delivery systems; see Sections 3.7 and 5.

If an SMTP server has accepted the task of relaying the mail and later finds that the destination is incorrect or that the mail cannot be delivered for some other reason, then it MUST construct an "undeliverable mail" notification message and send it to the originator of the undeliverable mail (as indicated by the reverse-path). Formats specified for non-delivery reports by other standards (see, for example, RFC 3461 [32] and RFC 3464 [33]) SHOULD be used if possible.

This notification message must be from the SMTP server at the relay host or the host that first determines that delivery cannot be accomplished. Of course, SMTP servers MUST NOT send notification messages about problems transporting notification messages. One way to prevent loops in error reporting is to specify a null reverse-path in the MAIL command of a notification message. When such a message is transmitted, the reverse-path MUST be set to null (see Section 4.5.5 for additional discussion). A MAIL command with a null reverse-path appears as follows:

```
MAIL FROM:<>
```

As discussed in Section 6.4, a relay SMTP has no need to inspect or act upon the header section or body of the message data and MUST NOT do so except to add its own "Received:" header field (Section 4.4) and, optionally, to attempt to detect looping in the mail system (see Section 6.3). Of course, this prohibition also applies to any modifications of these header fields or text (see also Section 7.9).

3.7. Mail Gatewaying

While the relay function discussed above operates within the Internet SMTP transport service environment, MX records or various forms of explicit routing may require that an intermediate SMTP server perform a translation function between one transport service and another. As discussed in Section 2.3.10, when such a system is at the boundary between two transport service environments, we refer to it as a "gateway" or "gateway SMTP".

Gatewaying mail between different mail environments, such as

different mail formats and protocols, is complex and does not easily yield to standardization. However, some general requirements may be given for a gateway between the Internet and another mail environment.

3.7.1. Header Fields in Gatewaying

Header fields MAY be rewritten when necessary as messages are gatewayed across mail environment boundaries. This may involve inspecting the message body or interpreting the local-part of the destination address in spite of the prohibitions in Section 6.4.

Other mail systems gatewayed to the Internet often use a subset of the RFC 822 header section or provide similar functionality with a different syntax, but some of these mail systems do not have an equivalent to the SMTP envelope. Therefore, when a message leaves the Internet environment, it may be necessary to fold the SMTP envelope information into the message header section. A possible solution would be to create new header fields to carry the envelope information (e.g., "X-SMTP-MAIL:" and "X-SMTP-RCPT:"); however, this would require changes in mail programs in foreign environments and might risk disclosure of private information (see Section 7.2).

3.7.2. Received Lines in Gatewaying

When forwarding a message into or out of the Internet environment, a gateway MUST prepend a Received: line, but it MUST NOT alter in any way a Received: line that is already in the header section.

"Received:" header fields of messages originating from other environments may not conform exactly to this specification. However, the most important use of Received: lines is for debugging mail faults, and this debugging can be severely hampered by well-meaning gateways that try to "fix" a Received: line. As another consequence of trace header fields arising in non-SMTP environments, receiving systems MUST NOT reject mail based on the format of a trace header field and SHOULD be extremely robust in the light of unexpected information or formats in those header fields.

The gateway SHOULD indicate the environment and protocol in the "via" clauses of Received header field(s) that it supplies.

3.7.3. Addresses in Gatewaying

From the Internet side, the gateway SHOULD accept all valid address formats in SMTP commands and in the RFC 822 header section, and all valid RFC 822 messages. Addresses and header fields generated by gateways MUST conform to applicable standards (including this one and RFC 5322 [4]). Gateways are, of course, subject to the same rules for handling source routes as those described for other SMTP systems in Section 3.3.

3.7.4. Other Header Fields in Gatewaying

The gateway MUST ensure that all header fields of a message that it forwards into the Internet mail environment meet the requirements for Internet mail. In particular, all addresses in "From:", "To:", "Cc:", etc., header fields MUST be transformed (if necessary) to satisfy the standard header syntax of RFC 5322 [4], MUST reference only fully-qualified domain names, and MUST be effective and useful for sending replies. The translation algorithm used to convert mail from the Internet protocols to another environment's protocol SHOULD ensure that error messages from the foreign mail environment are delivered to the reverse-path from the SMTP envelope, not to an address in the "From:", "Sender:", or similar header fields of the message.

3.7.5. Envelopes in Gatewaying

Similarly, when forwarding a message from another environment into the Internet, the gateway SHOULD set the envelope return path in accordance with an error message return address, if supplied by the foreign environment. If the foreign environment has no equivalent concept, the gateway must select and use a best approximation, with the message originator's address as the default of last resort.

3.8. Terminating Sessions and Connections

An SMTP connection is terminated when the client sends a QUIT command. The server responds with a positive reply code, after which it closes the connection.

An SMTP server MUST NOT intentionally close the connection under normal operational circumstances (see Section 7.8) except:

- After receiving a QUIT command and responding with a 221 reply.
- After detecting the need to shut down the SMTP service and returning a 421 response code. This response code can be issued after the server receives any command or, if necessary, asynchronously from command receipt (on the assumption that the client will receive it after the next command is issued).
- After a timeout, as specified in Section 4.5.3.2, occurs waiting for the client to send a command or data.

In particular, a server that closes connections in response to commands that are not understood is in violation of this specification. Servers are expected to be tolerant of unknown commands, issuing a 500 reply and awaiting further instructions from the client.

An SMTP server that is forcibly shut down via external means SHOULD attempt to send a line containing a 421 response code to the SMTP client before exiting. The SMTP client will normally read the 421 response code after sending its next command.

SMTP clients that experience a connection close, reset, or other communications failure due to circumstances not under their control (in violation of the intent of this specification but sometimes unavoidable) SHOULD, to maintain the robustness of the mail system, treat the mail transaction as if a 451 response had been received and act accordingly.

3.9. Mailing Lists and Aliases

An SMTP-capable host SHOULD support both the alias and the list models of address expansion for multiple delivery. When a message is delivered or forwarded to each address of an expanded list form, the return address in the envelope ("MAIL FROM:") MUST be changed to be the address of a person or other entity who administers the list. However, in this case, the message header section (RFC 5322 [4]) MUST be left unchanged; in particular, the "From" field of the header section is unaffected.

An important mail facility is a mechanism for multi-destination delivery of a single message, by transforming (or "expanding" or "exploding") a pseudo-mailbox address into a list of destination mailbox addresses. When a message is sent to such a pseudo-mailbox (sometimes called an "exploder"), copies are forwarded or redistributed to each mailbox in the expanded list. Servers SHOULD simply utilize the addresses on the list; application of heuristics or other matching rules to eliminate some addresses, such as that of the originator, is strongly discouraged. We classify such a pseudo-mailbox as an "alias" or a "list", depending upon the expansion rules.

3.9.1. Alias

To expand an alias, the recipient mailer simply replaces the pseudo-mailbox address in the envelope with each of the expanded addresses in turn; the rest of the envelope and the message body are left unchanged. The message is then delivered or forwarded to each expanded address.

3.9.2. List

A mailing list may be said to operate by "redistribution" rather than by "forwarding". To expand a list, the recipient mailer replaces the pseudo-mailbox address in the envelope with each of the expanded addresses in turn. The return (backward-pointing) address in the envelope is changed so that all error messages generated by the final deliveries will be returned to a list administrator, not to the message originator, who generally has no control over the contents of the list and will typically find error messages annoying. Note that the key difference between handling aliases (Section 3.9.1) and forwarding (this subsection) is the change to the backward-pointing address in this case. When a list constrains its processing to the very limited set of modifications and actions described here, it is attempting to emulate an MTA; such lists can be treated as a continuation in email transit.

There exist mailing lists that perform additional, sometimes extensive, modifications to a message and its envelope. Such mailing lists need to be viewed as full MUAs, which accept a delivery and post a new message.

4. The SMTP Specifications

4.1. SMTP Commands

4.1.1. Command Semantics and Syntax

The SMTP commands define the mail transfer or the mail system function requested by the user. SMTP commands are character strings terminated by <CRLF>. The commands themselves are alphabetic characters terminated by <SP> if parameters follow and <CRLF> otherwise. (In the interest of improved interoperability, SMTP receivers SHOULD tolerate trailing white space before the terminating <CRLF>.) The syntax of the local part of a mailbox MUST conform to receiver site conventions and the syntax specified in Section 4.1.2. The SMTP commands are discussed below. The SMTP replies are discussed in Section 4.2.

A mail transaction involves several data objects that are communicated as arguments to different commands. The reverse-path is the argument of the MAIL command, the forward-path is the argument of the RCPT command, and the mail data is the argument of the DATA command. These arguments or data objects must be transmitted and held, pending the confirmation communicated by the end of mail data indication that finalizes the transaction. The model for this is that distinct buffers are provided to hold the types of data objects; that is, there is a reverse-path buffer, a forward-path buffer, and a mail data buffer. Specific commands cause information to be appended to a specific buffer, or cause one or more buffers to be cleared.

Several commands (RSET, DATA, QUIT) are specified as not permitting parameters. In the absence of specific extensions offered by the server and accepted by the client, clients MUST NOT send such parameters and servers SHOULD reject commands containing them as having invalid syntax.

4.1.1.1. Extended HELLO (EHLO) or HELLO (HELO)

These commands are used to identify the SMTP client to the SMTP server. The argument clause contains the fully-qualified domain name of the SMTP client, if one is available. In situations in which the SMTP client system does not have a meaningful domain name (e.g., when its address is dynamically allocated and no reverse mapping record is available), the client SHOULD send an address literal (see Section 4.1.3).

RFC 2821, and some earlier informal practices, encouraged following the literal by information that would help to identify the client system. That convention was not widely supported, and many SMTP servers considered it an error. In the interest of interoperability, it is probably wise for servers to be prepared for this string to occur, but SMTP clients SHOULD NOT send it.

The SMTP server identifies itself to the SMTP client in the connection greeting reply and in the response to this command.

A client SMTP SHOULD start an SMTP session by issuing the EHLO command. If the SMTP server supports the SMTP service extensions, it will give a successful response, a failure response, or an error response. If the SMTP server, in violation of this specification, does not support any SMTP service extensions, it will generate an error response. Older client SMTP systems MAY, as discussed above, use HELO (as specified in RFC 821) instead of EHLO, and servers MUST support the HELO command and reply properly to it. In any event, a client MUST issue HELO or EHLO before starting a mail transaction.

These commands, and a "250 OK" reply to one of them, confirm that both the SMTP client and the SMTP server are in the initial state, that is, there is no transaction in progress and all state tables and buffers are cleared.

Syntax:

```
ehlo            = "EHLO" SP ( Domain / address-literal ) CRLF

helo            = "HELO" SP Domain CRLF
```

Normally, the response to EHLO will be a multiline reply. Each line of the response contains a keyword and, optionally, one or more parameters. Following the normal syntax for multiline replies, these keywords follow the code (250) and a hyphen for all but the last line, and the code and a space for the last line. The syntax for a positive response, using the ABNF notation and terminal symbols of RFC 5234 [7], is:

```
ehlo-ok-rsp     = ( "250" SP Domain [ SP ehlo-greet ] CRLF )
                  / ( "250-" Domain [ SP ehlo-greet ] CRLF
                  *( "250-" ehlo-line CRLF )
                  "250" SP ehlo-line CRLF )

ehlo-greet      = 1*(%d0-9 / %d11-12 / %d14-127)
                  ; string of any characters other than CR or LF

ehlo-line       = ehlo-keyword *( SP ehlo-param )

ehlo-keyword    = (ALPHA / DIGIT) *(ALPHA / DIGIT / "-")
                  ; additional syntax of ehlo-params depends on
                  ; ehlo-keyword

ehlo-param      = 1*(%d33-126)
                  ; any CHAR excluding <SP> and all
                  ; control characters (US-ASCII 0-31 and 127
                  ; inclusive)
```

Although EHLO keywords may be specified in upper, lower, or mixed case, they MUST always be recognized and processed in a case-

insensitive manner. This is simply an extension of practices specified in RFC 821 and Section 2.4.

The EHLO response MUST contain keywords (and associated parameters if required) for all commands not listed as "required" in Section 4.5.1 excepting only private-use commands as described in Section 4.1.5. Private-use commands MAY be listed.

4.1.1.2. MAIL (MAIL)

This command is used to initiate a mail transaction in which the mail data is delivered to an SMTP server that may, in turn, deliver it to one or more mailboxes or pass it on to another system (possibly using SMTP). The argument clause contains a reverse-path and may contain optional parameters. In general, the MAIL command may be sent only when no mail transaction is in progress, see Section 4.1.4.

The reverse-path consists of the sender mailbox. Historically, that mailbox might optionally have been preceded by a list of hosts, but that behavior is now deprecated (see Appendix C). In some types of reporting messages for which a reply is likely to cause a mail loop (for example, mail delivery and non-delivery notifications), the reverse-path may be null (see Section 3.6).

This command clears the reverse-path buffer, the forward-path buffer, and the mail data buffer, and it inserts the reverse-path information from its argument clause into the reverse-path buffer.

If service extensions were negotiated, the MAIL command may also carry parameters associated with a particular service extension.

Syntax:

mail = "MAIL FROM:" Reverse-path
[SP Mail-parameters] CRLF

4.1.1.3. RECIPIENT (RCPT)

This command is used to identify an individual recipient of the mail data; multiple recipients are specified by multiple uses of this command. The argument clause contains a forward-path and may contain optional parameters.

The forward-path normally consists of the required destination mailbox. Sending systems SHOULD NOT generate the optional list of hosts known as a source route. Receiving systems MUST recognize source route syntax but SHOULD strip off the source route specification and utilize the domain name associated with the mailbox as if the source route had not been provided.

Similarly, relay hosts SHOULD strip or ignore source routes, and names MUST NOT be copied into the reverse-path. When mail reaches its ultimate destination (the forward-path contains only a destination mailbox), the SMTP server inserts it into the destination mailbox in accordance with its host mail conventions.

This command appends its forward-path argument to the forward-path buffer; it does not change the reverse-path buffer nor the mail data buffer.

For example, mail received at relay host xyz.com with envelope commands

MAIL FROM:<userx@y.foo.org>
RCPT TO:<@hosta.int,@jkl.org:userc@d.bar.org>

will normally be sent directly on to host d.bar.org with envelope commands

```
   MAIL FROM:<userx@y.foo.org>
   RCPT TO:<userc@d.bar.org>
```

As provided in Appendix C, xyz.com MAY also choose to relay the message to hosta.int, using the envelope commands

```
   MAIL FROM:<userx@y.foo.org>
   RCPT TO:<@hosta.int,@jkl.org:userc@d.bar.org>
```

or to jkl.org, using the envelope commands

```
   MAIL FROM:<userx@y.foo.org>
   RCPT TO:<@jkl.org:userc@d.bar.org>
```

Attempting to use relaying this way is now strongly discouraged. Since hosts are not required to relay mail at all, xyz.com MAY also reject the message entirely when the RCPT command is received, using a 550 code (since this is a "policy reason").

If service extensions were negotiated, the RCPT command may also carry parameters associated with a particular service extension offered by the server. The client MUST NOT transmit parameters other than those associated with a service extension offered by the server in its EHLO response.

Syntax:

```
   rcpt = "RCPT TO:" ( "<Postmaster@" Domain ">" / "<Postmaster>" /
               Forward-path ) [SP Rcpt-parameters] CRLF

               Note that, in a departure from the usual rules for
               local-parts, the "Postmaster" string shown above is
               treated as case-insensitive.
```

4.1.1.4. DATA (DATA)

The receiver normally sends a 354 response to DATA, and then treats the lines (strings ending in <CRLF> sequences, as described in Section 2.3.7) following the command as mail data from the sender. This command causes the mail data to be appended to the mail data buffer. The mail data may contain any of the 128 ASCII character codes, although experience has indicated that use of control characters other than SP, HT, CR, and LF may cause problems and SHOULD be avoided when possible.

The mail data are terminated by a line containing only a period, that is, the character sequence "<CRLF>.<CRLF>", where the first <CRLF> is actually the terminator of the previous line (see Section 4.5.2). This is the end of mail data indication. The first <CRLF> of this terminating sequence is also the <CRLF> that ends the final line of the data (message text) or, if there was no mail data, ends the DATA command itself (the "no mail data" case does not conform to this specification since it would require that neither the trace header fields required by this specification nor the message header section required by RFC 5322 [4] be transmitted). An extra <CRLF> MUST NOT be added, as that would cause an empty line to be added to the message. The only exception to this rule would arise if the message body were passed to the originating SMTP-sender with a final "line" that did not end in <CRLF>; in that case, the originating SMTP system MUST either reject the message as invalid or add <CRLF> in order to have the receiving SMTP server recognize the "end of data" condition.

The custom of accepting lines ending only in <LF>, as a concession to non-conforming behavior on the part of some UNIX systems, has proven to cause more interoperability problems than it solves, and SMTP server systems MUST NOT do this, even in the name of improved robustness. In particular, the sequence "<LF>.<LF>" (bare line

feeds, without carriage returns) MUST NOT be treated as equivalent to <CRLF>.<CRLF> as the end of mail data indication.

Receipt of the end of mail data indication requires the server to process the stored mail transaction information. This processing consumes the information in the reverse-path buffer, the forward-path buffer, and the mail data buffer, and on the completion of this command these buffers are cleared. If the processing is successful, the receiver MUST send an OK reply. If the processing fails, the receiver MUST send a failure reply. The SMTP model does not allow for partial failures at this point: either the message is accepted by the server for delivery and a positive response is returned or it is not accepted and a failure reply is returned. In sending a positive "250 OK" completion reply to the end of data indication, the receiver takes full responsibility for the message (see Section 6.1). Errors that are diagnosed subsequently MUST be reported in a mail message, as discussed in Section 4.4.

When the SMTP server accepts a message either for relaying or for final delivery, it inserts a trace record (also referred to interchangeably as a "time stamp line" or "Received" line) at the top of the mail data. This trace record indicates the identity of the host that sent the message, the identity of the host that received the message (and is inserting this time stamp), and the date and time the message was received. Relayed messages will have multiple time stamp lines. Details for formation of these lines, including their syntax, is specified in Section 4.4.

Additional discussion about the operation of the DATA command appears in Section 3.3.

Syntax:

```
data = "DATA" CRLF
```

4.1.1.5. RESET (RSET)

This command specifies that the current mail transaction will be aborted. Any stored sender, recipients, and mail data MUST be discarded, and all buffers and state tables cleared. The receiver MUST send a "250 OK" reply to a RSET command with no arguments. A reset command may be issued by the client at any time. It is effectively equivalent to a NOOP (i.e., it has no effect) if issued immediately after EHLO, before EHLO is issued in the session, after an end of data indicator has been sent and acknowledged, or immediately before a QUIT. An SMTP server MUST NOT close the connection as the result of receiving a RSET; that action is reserved for QUIT (see Section 4.1.1.10).

Since EHLO implies some additional processing and response by the server, RSET will normally be more efficient than reissuing that command, even though the formal semantics are the same.

There are circumstances, contrary to the intent of this specification, in which an SMTP server may receive an indication that the underlying TCP connection has been closed or reset. To preserve the robustness of the mail system, SMTP servers SHOULD be prepared for this condition and SHOULD treat it as if a QUIT had been received before the connection disappeared.

Syntax:

```
rset = "RSET" CRLF
```

4.1.1.6. VERIFY (VRFY)

This command asks the receiver to confirm that the argument identifies a user or mailbox. If it is a user name, information is

returned as specified in Section 3.5.

This command has no effect on the reverse-path buffer, the forward-path buffer, or the mail data buffer.

Syntax:

vrfy = "VRFY" SP String CRLF

4.1.1.7. EXPAND (EXPN)

This command asks the receiver to confirm that the argument identifies a mailing list, and if so, to return the membership of that list. If the command is successful, a reply is returned containing information as described in Section 3.5. This reply will have multiple lines except in the trivial case of a one-member list.

This command has no effect on the reverse-path buffer, the forward-path buffer, or the mail data buffer, and it may be issued at any time.

Syntax:

expn = "EXPN" SP String CRLF

4.1.1.8. HELP (HELP)

This command causes the server to send helpful information to the client. The command MAY take an argument (e.g., any command name) and return more specific information as a response.

This command has no effect on the reverse-path buffer, the forward-path buffer, or the mail data buffer, and it may be issued at any time.

SMTP servers SHOULD support HELP without arguments and MAY support it with arguments.

Syntax:

help = "HELP" [SP String] CRLF

4.1.1.9. NOOP (NOOP)

This command does not affect any parameters or previously entered commands. It specifies no action other than that the receiver send a "250 OK" reply.

This command has no effect on the reverse-path buffer, the forward-path buffer, or the mail data buffer, and it may be issued at any time. If a parameter string is specified, servers SHOULD ignore it.

Syntax:

noop = "NOOP" [SP String] CRLF

4.1.1.10. QUIT (QUIT)

This command specifies that the receiver MUST send a "221 OK" reply, and then close the transmission channel.

The receiver MUST NOT intentionally close the transmission channel until it receives and replies to a QUIT command (even if there was an error). The sender MUST NOT intentionally close the transmission channel until it sends a QUIT command, and it SHOULD wait until it receives the reply (even if there was an error response to a previous command). If the connection is closed prematurely due to violations

of the above or system or network failure, the server MUST cancel any pending transaction, but not undo any previously completed transaction, and generally MUST act as if the command or transaction in progress had received a temporary error (i.e., a 4yz response).

The QUIT command may be issued at any time. Any current uncompleted mail transaction will be aborted.

Syntax:

```
quit = "QUIT" CRLF
```

4.1.1.11. Mail-Parameter and Rcpt-Parameter Error Responses

If the server SMTP does not recognize or cannot implement one or more of the parameters associated with a particular MAIL FROM or RCPT TO command, it will return code 555.

If, for some reason, the server is temporarily unable to accommodate one or more of the parameters associated with a MAIL FROM or RCPT TO command, and if the definition of the specific parameter does not mandate the use of another code, it should return code 455.

Errors specific to particular parameters and their values will be specified in the parameter's defining RFC.

4.1.2. Command Argument Syntax

The syntax of the argument clauses of the above commands (using the syntax specified in RFC 5234 [7] where applicable) is given below. Some of the productions given below are used only in conjunction with source routes as described in Appendix C. Terminals not defined in this document, such as ALPHA, DIGIT, SP, CR, LF, CRLF, are as defined in the "core" syntax in Section 6 of RFC 5234 [7] or in the message format syntax in RFC 5322 [4].

```
Reverse-path   = Path / "<>"

Forward-path   = Path

Path           = "<" [ A-d-l ":" ] Mailbox ">"

A-d-l          = At-domain *( "," At-domain )
               ; Note that this form, the so-called "source
               ; route", MUST BE accepted, SHOULD NOT be
               ; generated, and SHOULD be ignored.

At-domain      = "@" Domain

Mail-parameters  = esmtp-param *(SP esmtp-param)

Rcpt-parameters  = esmtp-param *(SP esmtp-param)

esmtp-param    = esmtp-keyword ["=" esmtp-value]

esmtp-keyword  = (ALPHA / DIGIT) *(ALPHA / DIGIT / "-")

esmtp-value    = 1*(%d33-60 / %d62-126)
               ; any CHAR excluding "=", SP, and control
               ; characters.  If this string is an email address,
               ; i.e., a Mailbox, then the "xtext" syntax [32]
               ; SHOULD be used.

Keyword        = Ldh-str

Argument       = Atom

Domain         = sub-domain *("." sub-domain)
```

```
sub-domain     = Let-dig [Ldh-str]

Let-dig        = ALPHA / DIGIT

Ldh-str        = *( ALPHA / DIGIT / "-" ) Let-dig

address-literal  = "[" ( IPv4-address-literal /
                 IPv6-address-literal /
                 General-address-literal ) "]"
                 ; See Section 4.1.3

Mailbox        = Local-part "@" ( Domain / address-literal )

Local-part     = Dot-string / Quoted-string
               ; MAY be case-sensitive

Dot-string     = Atom *("."  Atom)

Atom           = 1*atext

Quoted-string  = DQUOTE *QcontentSMTP DQUOTE

QcontentSMTP   = qtextSMTP / quoted-pairSMTP

quoted-pairSMTP  = %d92 %d32-126
                 ; i.e., backslash followed by any ASCII
                 ; graphic (including itself) or SPace

qtextSMTP      = %d32-33 / %d35-91 / %d93-126
               ; i.e., within a quoted string, any
               ; ASCII graphic or space is permitted
               ; without blackslash-quoting except
               ; double-quote and the backslash itself.

String         = Atom / Quoted-string
```

While the above definition for Local-part is relatively permissive, for maximum interoperability, a host that expects to receive mail SHOULD avoid defining mailboxes where the Local-part requires (or uses) the Quoted-string form or where the Local-part is case-sensitive. For any purposes that require generating or comparing Local-parts (e.g., to specific mailbox names), all quoted forms MUST be treated as equivalent, and the sending system SHOULD transmit the form that uses the minimum quoting possible.

Systems MUST NOT define mailboxes in such a way as to require the use in SMTP of non-ASCII characters (octets with the high order bit set to one) or ASCII "control characters" (decimal value 0-31 and 127). These characters MUST NOT be used in MAIL or RCPT commands or other commands that require mailbox names.

Note that the backslash, "\", is a quote character, which is used to indicate that the next character is to be used literally (instead of its normal interpretation). For example, "Joe\,Smith" indicates a single nine-character user name string with the comma being the fourth character of that string.

To promote interoperability and consistent with long-standing guidance about conservative use of the DNS in naming and applications (e.g., see Section 2.3.1 of the base DNS document, RFC 1035 [2]), characters outside the set of alphabetic characters, digits, and hyphen MUST NOT appear in domain name labels for SMTP clients or servers. In particular, the underscore character is not permitted. SMTP servers that receive a command in which invalid character codes have been employed, and for which there are no other reasons for rejection, MUST reject that command with a 501 response (this rule, like others, could be overridden by appropriate SMTP extensions).

4.1.3. Address Literals

Sometimes a host is not known to the domain name system and communication (and, in particular, communication to report and repair the error) is blocked. To bypass this barrier, a special literal form of the address is allowed as an alternative to a domain name. For IPv4 addresses, this form uses four small decimal integers separated by dots and enclosed by brackets such as [123.255.37.2], which indicates an (IPv4) Internet Address in sequence-of-octets form. For IPv6 and other forms of addressing that might eventually be standardized, the form consists of a standardized "tag" that identifies the address syntax, a colon, and the address itself, in a format specified as part of the relevant standards (i.e., RFC 4291 [8] for IPv6).

Specifically:

```
IPv4-address-literal  = Snum 3("."  Snum)

IPv6-address-literal  = "IPv6:" IPv6-addr

General-address-literal  = Standardized-tag ":" 1*dcontent

Standardized-tag  = Ldh-str
                  ; Standardized-tag MUST be specified in a
                  ; Standards-Track RFC and registered with IANA

dcontent       = %d33-90 / ; Printable US-ASCII
                 %d94-126 ; excl. "[", "\", "]"

Snum           = 1*3DIGIT
                 ; representing a decimal integer
                 ; value in the range 0 through 255

IPv6-addr      = IPv6-full / IPv6-comp / IPv6v4-full / IPv6v4-comp

IPv6-hex       = 1*4HEXDIG

IPv6-full      = IPv6-hex 7(":" IPv6-hex)

IPv6-comp      = [IPv6-hex *5(":" IPv6-hex)] "::"
                 [IPv6-hex *5(":" IPv6-hex)]
                 ; The "::" represents at least 2 16-bit groups of
                 ; zeros.  No more than 6 groups in addition to the
                 ; "::" may be present.

IPv6v4-full    = IPv6-hex 5(":" IPv6-hex) ":" IPv4-address-literal

IPv6v4-comp    = [IPv6-hex *3(":" IPv6-hex)] "::"
                 [IPv6-hex *3(":" IPv6-hex) ":"]
                 IPv4-address-literal
                 ; The "::" represents at least 2 16-bit groups of
                 ; zeros.  No more than 4 groups in addition to the
                 ; "::" and IPv4-address-literal may be present.
```

4.1.4. Order of Commands

There are restrictions on the order in which these commands may be used.

A session that will contain mail transactions MUST first be initialized by the use of the EHLO command. An SMTP server SHOULD accept commands for non-mail transactions (e.g., VRFY or EXPN) without this initialization.

An EHLO command MAY be issued by a client later in the session. If it is issued after the session begins and the EHLO command is

acceptable to the SMTP server, the SMTP server MUST clear all buffers and reset the state exactly as if a RSET command had been issued. In other words, the sequence of RSET followed immediately by EHLO is redundant, but not harmful other than in the performance cost of executing unnecessary commands.

If the EHLO command is not acceptable to the SMTP server, 501, 500, 502, or 550 failure replies MUST be returned as appropriate. The SMTP server MUST stay in the same state after transmitting these replies that it was in before the EHLO was received.

The SMTP client MUST, if possible, ensure that the domain parameter to the EHLO command is a primary host name as specified for this command in Section 2.3.5. If this is not possible (e.g., when the client's address is dynamically assigned and the client does not have an obvious name), an address literal SHOULD be substituted for the domain name.

An SMTP server MAY verify that the domain name argument in the EHLO command actually corresponds to the IP address of the client. However, if the verification fails, the server MUST NOT refuse to accept a message on that basis. Information captured in the verification attempt is for logging and tracing purposes. Note that this prohibition applies to the matching of the parameter to its IP address only; see Section 7.9 for a more extensive discussion of rejecting incoming connections or mail messages.

The NOOP, HELP, EXPN, VRFY, and RSET commands can be used at any time during a session, or without previously initializing a session. SMTP servers SHOULD process these normally (that is, not return a 503 code) even if no EHLO command has yet been received; clients SHOULD open a session with EHLO before sending these commands.

If these rules are followed, the example in RFC 821 that shows "550 access denied to you" in response to an EXPN command is incorrect unless an EHLO command precedes the EXPN or the denial of access is based on the client's IP address or other authentication or authorization-determining mechanisms.

The MAIL command (or the obsolete SEND, SOML, or SAML commands) begins a mail transaction. Once started, a mail transaction consists of a transaction beginning command, one or more RCPT commands, and a DATA command, in that order. A mail transaction may be aborted by the RSET, a new EHLO, or the QUIT command. There may be zero or more transactions in a session. MAIL (or SEND, SOML, or SAML) MUST NOT be sent if a mail transaction is already open, i.e., it should be sent only if no mail transaction had been started in the session, or if the previous one successfully concluded with a successful DATA command, or if the previous one was aborted, e.g., with a RSET or new EHLO.

If the transaction beginning command argument is not acceptable, a 501 failure reply MUST be returned and the SMTP server MUST stay in the same state. If the commands in a transaction are out of order to the degree that they cannot be processed by the server, a 503 failure reply MUST be returned and the SMTP server MUST stay in the same state.

The last command in a session MUST be the QUIT command. The QUIT command SHOULD be used by the client SMTP to request connection closure, even when no session opening command was sent and accepted.

4.1.5. Private-Use Commands

As specified in Section 2.2.2, commands starting in "X" may be used by bilateral agreement between the client (sending) and server (receiving) SMTP agents. An SMTP server that does not recognize such a command is expected to reply with "500 Command not recognized". An

extended SMTP server MAY list the feature names associated with these private commands in the response to the EHLO command.

Commands sent or accepted by SMTP systems that do not start with "X" MUST conform to the requirements of Section 2.2.2.

4.2. SMTP Replies

Replies to SMTP commands serve to ensure the synchronization of requests and actions in the process of mail transfer and to guarantee that the SMTP client always knows the state of the SMTP server. Every command MUST generate exactly one reply.

The details of the command-reply sequence are described in Section 4.3.

An SMTP reply consists of a three digit number (transmitted as three numeric characters) followed by some text unless specified otherwise in this document. The number is for use by automata to determine what state to enter next; the text is for the human user. The three digits contain enough encoded information that the SMTP client need not examine the text and may either discard it or pass it on to the user, as appropriate. Exceptions are as noted elsewhere in this document. In particular, the 220, 221, 251, 421, and 551 reply codes are associated with message text that must be parsed and interpreted by machines. In the general case, the text may be receiver dependent and context dependent, so there are likely to be varying texts for each reply code. A discussion of the theory of reply codes is given in Section 4.2.1. Formally, a reply is defined to be the sequence: a three-digit code, <SP>, one line of text, and <CRLF>, or a multiline reply (as defined in the same section). Since, in violation of this specification, the text is sometimes not sent, clients that do not receive it SHOULD be prepared to process the code alone (with or without a trailing space character). Only the EHLO, EXPN, and HELP commands are expected to result in multiline replies in normal circumstances; however, multiline replies are allowed for any command.

In ABNF, server responses are:

```
Greeting       = ( "220 " (Domain / address-literal)
                 [ SP textstring ] CRLF ) /
                 ( "220-" (Domain / address-literal)
                 [ SP textstring ] CRLF
                 *( "220-" [ textstring ] CRLF )
                 "220" [ SP textstring ] CRLF )

textstring     = 1*(%d09 / %d32-126) ; HT, SP, Printable US-ASCII

Reply-line     = *( Reply-code "-" [ textstring ] CRLF )
                 Reply-code [ SP textstring ] CRLF

Reply-code     = %x32-35 %x30-35 %x30-39
```

where "Greeting" appears only in the 220 response that announces that the server is opening its part of the connection. (Other possible server responses upon connection follow the syntax of Reply-line.)

An SMTP server SHOULD send only the reply codes listed in this document. An SMTP server SHOULD use the text shown in the examples whenever appropriate.

An SMTP client MUST determine its actions only by the reply code, not by the text (except for the "change of address" 251 and 551 and, if necessary, 220, 221, and 421 replies); in the general case, any text, including no text at all (although senders SHOULD NOT send bare codes), MUST be acceptable. The space (blank) following the reply code is considered part of the text. Whenever possible, a receiver-

SMTP SHOULD test the first digit (severity indication) of the reply code.

The list of codes that appears below MUST NOT be construed as permanent. While the addition of new codes should be a rare and significant activity, with supplemental information in the textual part of the response being preferred, new codes may be added as the result of new Standards or Standards-Track specifications. Consequently, a sender-SMTP MUST be prepared to handle codes not specified in this document and MUST do so by interpreting the first digit only.

In the absence of extensions negotiated with the client, SMTP servers MUST NOT send reply codes whose first digits are other than 2, 3, 4, or 5. Clients that receive such out-of-range codes SHOULD normally treat them as fatal errors and terminate the mail transaction.

4.2.1. Reply Code Severities and Theory

The three digits of the reply each have a special significance. The first digit denotes whether the response is good, bad, or incomplete. An unsophisticated SMTP client, or one that receives an unexpected code, will be able to determine its next action (proceed as planned, redo, retrench, etc.) by examining this first digit. An SMTP client that wants to know approximately what kind of error occurred (e.g., mail system error, command syntax error) may examine the second digit. The third digit and any supplemental information that may be present is reserved for the finest gradation of information.

There are four values for the first digit of the reply code:

2yz Positive Completion reply
The requested action has been successfully completed. A new request may be initiated.

3yz Positive Intermediate reply
The command has been accepted, but the requested action is being held in abeyance, pending receipt of further information. The SMTP client should send another command specifying this information. This reply is used in command sequence groups (i.e., in DATA).

4yz Transient Negative Completion reply
The command was not accepted, and the requested action did not occur. However, the error condition is temporary, and the action may be requested again. The sender should return to the beginning of the command sequence (if any). It is difficult to assign a meaning to "transient" when two different sites (receiver- and sender-SMTP agents) must agree on the interpretation. Each reply in this category might have a different time value, but the SMTP client SHOULD try again. A rule of thumb to determine whether a reply fits into the 4yz or the 5yz category (see below) is that replies are 4yz if they can be successful if repeated without any change in command form or in properties of the sender or receiver (that is, the command is repeated identically and the receiver does not put up a new implementation).

5yz Permanent Negative Completion reply
The command was not accepted and the requested action did not occur. The SMTP client SHOULD NOT repeat the exact request (in the same sequence). Even some "permanent" error conditions can be corrected, so the human user may want to direct the SMTP client to reinitiate the command sequence by direct action at some point in the future (e.g., after the spelling has been changed, or the user has altered the account status).

It is worth noting that the file transfer protocol (FTP) [34] uses a very similar code architecture and that the SMTP codes are based on

the FTP model. However, SMTP uses a one-command, one-response model (while FTP is asynchronous) and FTP's 1yz codes are not part of the SMTP model.

The second digit encodes responses in specific categories:

x0z Syntax: These replies refer to syntax errors, syntactically correct commands that do not fit any functional category, and unimplemented or superfluous commands.

x1z Information: These are replies to requests for information, such as status or help.

x2z Connections: These are replies referring to the transmission channel.

x3z Unspecified.

x4z Unspecified.

x5z Mail system: These replies indicate the status of the receiver mail system vis-a-vis the requested transfer or other mail system action.

The third digit gives a finer gradation of meaning in each category specified by the second digit. The list of replies illustrates this. Each reply text is recommended rather than mandatory, and may even change according to the command with which it is associated. On the other hand, the reply codes must strictly follow the specifications in this section. Receiver implementations should not invent new codes for slightly different situations from the ones described here, but rather adapt codes already defined.

For example, a command such as NOOP, whose successful execution does not offer the SMTP client any new information, will return a 250 reply. The reply is 502 when the command requests an unimplemented non-site-specific action. A refinement of that is the 504 reply for a command that is implemented, but that requests an unimplemented parameter.

The reply text may be longer than a single line; in these cases the complete text must be marked so the SMTP client knows when it can stop reading the reply. This requires a special format to indicate a multiple line reply.

The format for multiline replies requires that every line, except the last, begin with the reply code, followed immediately by a hyphen, "-" (also known as minus), followed by text. The last line will begin with the reply code, followed immediately by <SP>, optionally some text, and <CRLF>. As noted above, servers SHOULD send the <SP> if subsequent text is not sent, but clients MUST be prepared for it to be omitted.

For example:

```
250-First line
250-Second line
250-234 Text beginning with numbers
250 The last line
```

In a multiline reply, the reply code on each of the lines MUST be the same. It is reasonable for the client to rely on this, so it can make processing decisions based on the code in any line, assuming that all others will be the same. In a few cases, there is important data for the client in the reply "text". The client will be able to identify these cases from the current context.

4.2.2. Reply Codes by Function Groups

500 Syntax error, command unrecognized (This may include errors such as command line too long)

501 Syntax error in parameters or arguments

502 Command not implemented (see Section 4.2.4)

503 Bad sequence of commands

504 Command parameter not implemented

211 System status, or system help reply

214 Help message (Information on how to use the receiver or the meaning of a particular non-standard command; this reply is useful only to the human user)

220 <domain> Service ready

221 <domain> Service closing transmission channel

421 <domain> Service not available, closing transmission channel (This may be a reply to any command if the service knows it must shut down)

250 Requested mail action okay, completed

251 User not local; will forward to <forward-path> (See Section 3.4)

252 Cannot VRFY user, but will accept message and attempt delivery (See Section 3.5.3)

455 Server unable to accommodate parameters

555 MAIL FROM/RCPT TO parameters not recognized or not implemented

450 Requested mail action not taken: mailbox unavailable (e.g., mailbox busy or temporarily blocked for policy reasons)

550 Requested action not taken: mailbox unavailable (e.g., mailbox not found, no access, or command rejected for policy reasons)

451 Requested action aborted: error in processing

551 User not local; please try <forward-path> (See Section 3.4)

452 Requested action not taken: insufficient system storage

552 Requested mail action aborted: exceeded storage allocation

553 Requested action not taken: mailbox name not allowed (e.g., mailbox syntax incorrect)

354 Start mail input; end with <CRLF>.<CRLF>

554 Transaction failed (Or, in the case of a connection-opening response, "No SMTP service here")

4.2.3. Reply Codes in Numeric Order

211 System status, or system help reply

214 Help message (Information on how to use the receiver or the meaning of a particular non-standard command; this reply is useful only to the human user)

220 <domain> Service ready

221 <domain> Service closing transmission channel

250 Requested mail action okay, completed

251 User not local; will forward to <forward-path> (See Section 3.4)

252 Cannot VRFY user, but will accept message and attempt delivery (See Section 3.5.3)

354 Start mail input; end with <CRLF>.<CRLF>

421 <domain> Service not available, closing transmission channel (This may be a reply to any command if the service knows it must shut down)

450 Requested mail action not taken: mailbox unavailable (e.g., mailbox busy or temporarily blocked for policy reasons)

451 Requested action aborted: local error in processing

452 Requested action not taken: insufficient system storage

455 Server unable to accommodate parameters

500 Syntax error, command unrecognized (This may include errors such as command line too long)

501 Syntax error in parameters or arguments

502 Command not implemented (see Section 4.2.4)

503 Bad sequence of commands

504 Command parameter not implemented

550 Requested action not taken: mailbox unavailable (e.g., mailbox not found, no access, or command rejected for policy reasons)

551 User not local; please try <forward-path> (See Section 3.4)

552 Requested mail action aborted: exceeded storage allocation

553 Requested action not taken: mailbox name not allowed (e.g., mailbox syntax incorrect)

554 Transaction failed (Or, in the case of a connection-opening response, "No SMTP service here")

555 MAIL FROM/RCPT TO parameters not recognized or not implemented

4.2.4. Reply Code 502

Questions have been raised as to when reply code 502 (Command not implemented) SHOULD be returned in preference to other codes. 502 SHOULD be used when the command is actually recognized by the SMTP server, but not implemented. If the command is not recognized, code 500 SHOULD be returned. Extended SMTP systems MUST NOT list capabilities in response to EHLO for which they will return 502 (or 500) replies.

4.2.5. Reply Codes after DATA and the Subsequent <CRLF>.<CRLF>

When an SMTP server returns a positive completion status (2yz code) after the DATA command is completed with <CRLF>.<CRLF>, it accepts responsibility for:

o delivering the message (if the recipient mailbox exists), or

- o if attempts to deliver the message fail due to transient conditions, retrying delivery some reasonable number of times at intervals as specified in Section 4.5.4.

- o if attempts to deliver the message fail due to permanent conditions, or if repeated attempts to deliver the message fail due to transient conditions, returning appropriate notification to the sender of the original message (using the address in the SMTP MAIL command).

When an SMTP server returns a temporary error status (4yz) code after the DATA command is completed with <CRLF>.<CRLF>, it MUST NOT make a subsequent attempt to deliver that message. The SMTP client retains responsibility for the delivery of that message and may either return it to the user or requeue it for a subsequent attempt (see Section 4.5.4.1).

The user who originated the message SHOULD be able to interpret the return of a transient failure status (by mail message or otherwise) as a non-delivery indication, just as a permanent failure would be interpreted. If the client SMTP successfully handles these conditions, the user will not receive such a reply.

When an SMTP server returns a permanent error status (5yz) code after the DATA command is completed with <CRLF>.<CRLF>, it MUST NOT make any subsequent attempt to deliver the message. As with temporary error status codes, the SMTP client retains responsibility for the message, but SHOULD not again attempt delivery to the same server without user review of the message and response and appropriate intervention.

4.3. Sequencing of Commands and Replies

4.3.1. Sequencing Overview

The communication between the sender and receiver is an alternating dialogue, controlled by the sender. As such, the sender issues a command and the receiver responds with a reply. Unless other arrangements are negotiated through service extensions, the sender MUST wait for this response before sending further commands. One important reply is the connection greeting. Normally, a receiver will send a 220 "Service ready" reply when the connection is completed. The sender SHOULD wait for this greeting message before sending any commands.

Note: all the greeting-type replies have the official name (the fully-qualified primary domain name) of the server host as the first word following the reply code. Sometimes the host will have no meaningful name. See Section 4.1.3 for a discussion of alternatives in these situations.

For example,

```
220 ISIF.USC.EDU Service ready
```

or

```
220 mail.example.com SuperSMTP v 6.1.2 Service ready
```

or

```
220 [10.0.0.1] Clueless host service ready
```

The table below lists alternative success and failure replies for each command. These SHOULD be strictly adhered to. A receiver MAY substitute text in the replies, but the meanings and actions implied by the code numbers and by the specific command reply sequence MUST

be preserved.

4.3.2. Command-Reply Sequences

Each command is listed with its usual possible replies. The prefixes used before the possible replies are "I" for intermediate, "S" for success, and "E" for error. Since some servers may generate other replies under special circumstances, and to allow for future extension, SMTP clients SHOULD, when possible, interpret only the first digit of the reply and MUST be prepared to deal with unrecognized reply codes by interpreting the first digit only. Unless extended using the mechanisms described in Section 2.2, SMTP servers MUST NOT transmit reply codes to an SMTP client that are other than three digits or that do not start in a digit between 2 and 5 inclusive.

These sequencing rules and, in principle, the codes themselves, can be extended or modified by SMTP extensions offered by the server and accepted (requested) by the client. However, if the target is more precise granularity in the codes, rather than codes for completely new purposes, the system described in RFC 3463 [25] SHOULD be used in preference to the invention of new codes.

In addition to the codes listed below, any SMTP command can return any of the following codes if the corresponding unusual circumstances are encountered:

500 For the "command line too long" case or if the command name was not recognized. Note that producing a "command not recognized" error in response to the required subset of these commands is a violation of this specification. Similarly, producing a "command too long" message for a command line shorter than 512 characters would violate the provisions of Section 4.5.3.1.4.

501 Syntax error in command or arguments. In order to provide for future extensions, commands that are specified in this document as not accepting arguments (DATA, RSET, QUIT) SHOULD return a 501 message if arguments are supplied in the absence of EHLO-advertised extensions.

421 Service shutting down and closing transmission channel

Specific sequences are:

CONNECTION ESTABLISHMENT

S: 220
E: 554

EHLO or HELO

S: 250
E: 504 (a conforming implementation could return this code only in fairly obscure cases), 550, 502 (permitted only with an old-style server that does not support EHLO)

MAIL

S: 250
E: 552, 451, 452, 550, 553, 503, 455, 555

RCPT

S: 250, 251 (but see Section 3.4 for discussion of 251 and 551)
E: 550, 551, 552, 553, 450, 451, 452, 503, 455, 555

DATA

```
         I: 354 -> data -> S: 250

                           E: 552, 554, 451, 452

                           E: 450, 550 (rejections for policy reasons)

         E: 503, 554

      RSET

         S: 250

      VRFY

         S: 250, 251, 252
         E: 550, 551, 553, 502, 504

      EXPN

         S: 250, 252
         E: 550, 500, 502, 504

      HELP

         S: 211, 214
         E: 502, 504

      NOOP

         S: 250

      QUIT

         S: 221
```

4.4. Trace Information

When an SMTP server receives a message for delivery or further processing, it MUST insert trace ("time stamp" or "Received") information at the beginning of the message content, as discussed in Section 4.1.1.4.

This line MUST be structured as follows:

- The FROM clause, which MUST be supplied in an SMTP environment, SHOULD contain both (1) the name of the source host as presented in the EHLO command and (2) an address literal containing the IP address of the source, determined from the TCP connection.

- The ID clause MAY contain an "@" as suggested in RFC 822, but this is not required.

- If the FOR clause appears, it MUST contain exactly one <path> entry, even when multiple RCPT commands have been given. Multiple <path>s raise some security issues and have been deprecated, see Section 7.2.

An Internet mail program MUST NOT change or delete a Received: line that was previously added to the message header section. SMTP servers MUST prepend Received lines to messages; they MUST NOT change the order of existing lines or insert Received lines in any other location.

As the Internet grows, comparability of Received header fields is important for detecting problems, especially slow relays. SMTP servers that create Received header fields SHOULD use explicit offsets in the dates (e.g., -0800), rather than time zone names of any type. Local time (with an offset) SHOULD be used rather than UT

when feasible. This formulation allows slightly more information about local circumstances to be specified. If UT is needed, the receiver need merely do some simple arithmetic to convert the values. Use of UT loses information about the time zone-location of the server. If it is desired to supply a time zone name, it SHOULD be included in a comment.

When the delivery SMTP server makes the "final delivery" of a message, it inserts a return-path line at the beginning of the mail data. This use of return-path is required; mail systems MUST support it. The return-path line preserves the information in the <reverse-path> from the MAIL command. Here, final delivery means the message has left the SMTP environment. Normally, this would mean it had been delivered to the destination user or an associated mail drop, but in some cases it may be further processed and transmitted by another mail system.

It is possible for the mailbox in the return path to be different from the actual sender's mailbox, for example, if error responses are to be delivered to a special error handling mailbox rather than to the message sender. When mailing lists are involved, this arrangement is common and useful as a means of directing errors to the list maintainer rather than the message originator.

The text above implies that the final mail data will begin with a return path line, followed by one or more time stamp lines. These lines will be followed by the rest of the mail data: first the balance of the mail header section and then the body (RFC 5322 [4]).

It is sometimes difficult for an SMTP server to determine whether or not it is making final delivery since forwarding or other operations may occur after the message is accepted for delivery. Consequently, any further (forwarding, gateway, or relay) systems MAY remove the return path and rebuild the MAIL command as needed to ensure that exactly one such line appears in a delivered message.

A message-originating SMTP system SHOULD NOT send a message that already contains a Return-path header field. SMTP servers performing a relay function MUST NOT inspect the message data, and especially not to the extent needed to determine if Return-path header fields are present. SMTP servers making final delivery MAY remove Return-path header fields before adding their own.

The primary purpose of the Return-path is to designate the address to which messages indicating non-delivery or other mail system failures are to be sent. For this to be unambiguous, exactly one return path SHOULD be present when the message is delivered. Systems using RFC 822 syntax with non-SMTP transports SHOULD designate an unambiguous address, associated with the transport envelope, to which error reports (e.g., non-delivery messages) should be sent.

Historical note: Text in RFC 822 that appears to contradict the use of the Return-path header field (or the envelope reverse-path address from the MAIL command) as the destination for error messages is not applicable on the Internet. The reverse-path address (as copied into the Return-path) MUST be used as the target of any mail containing delivery error messages.

In particular:

- a gateway from SMTP -> elsewhere SHOULD insert a return-path header field, unless it is known that the "elsewhere" transport also uses Internet domain addresses and maintains the envelope sender address separately.

- a gateway from elsewhere -> SMTP SHOULD delete any return-path header field present in the message, and either copy that information to the SMTP envelope or combine it with information present in the envelope of the other transport system to construct

the reverse-path argument to the MAIL command in the SMTP envelope.

The server must give special treatment to cases in which the processing following the end of mail data indication is only partially successful. This could happen if, after accepting several recipients and the mail data, the SMTP server finds that the mail data could be successfully delivered to some, but not all, of the recipients. In such cases, the response to the DATA command MUST be an OK reply. However, the SMTP server MUST compose and send an "undeliverable mail" notification message to the originator of the message.

A single notification listing all of the failed recipients or separate notification messages MUST be sent for each failed recipient. For economy of processing by the sender, the former SHOULD be used when possible. Note that the key difference between handling aliases (Section 3.9.1) and forwarding (this subsection) is the change to the backward-pointing address in this case. All notification messages about undeliverable mail MUST be sent using the MAIL command (even if they result from processing the obsolete SEND, SOML, or SAML commands) and MUST use a null return path as discussed in Section 3.6.

The time stamp line and the return path line are formally defined as follows (the definitions for "FWS" and "CFWS" appear in RFC 5322 [4]):

```
Return-path-line  = "Return-Path:" FWS Reverse-path <CRLF>

Time-stamp-line   = "Received:" FWS Stamp <CRLF>

Stamp             = From-domain By-domain Opt-info [CFWS] ";"
                    FWS date-time
                    ; where "date-time" is as defined in RFC 5322 [4]
                    ; but the "obs-" forms, especially two-digit
                    ; years, are prohibited in SMTP and MUST NOT be used.

From-domain       = "FROM" FWS Extended-Domain

By-domain         = CFWS "BY" FWS Extended-Domain

Extended-Domain   = Domain /
                    ( Domain FWS "(" TCP-info ")" ) /
                    ( address-literal FWS "(" TCP-info ")" )

TCP-info          = address-literal / ( Domain FWS address-literal )
                    ; Information derived by server from TCP connection
                    ; not client EHLO.

Opt-info          = [Via] [With] [ID] [For]
                    [Additional-Registered-Clauses]

Via               = CFWS "VIA" FWS Link

With              = CFWS "WITH" FWS Protocol

ID                = CFWS "ID" FWS ( Atom / msg-id )
                    ; msg-id is defined in RFC 5322 [4]

For               = CFWS "FOR" FWS ( Path / Mailbox )

Additional-Registered-Clauses  = CFWS Atom FWS String
                                 ; Additional standard clauses may be
                                 added in this
                                 ; location by future standards and
                                 registration with
                                 ; IANA.  SMTP servers SHOULD NOT use
```

```
                                    unregistered
                                    ; names.  See Section 8.

   Link            = "TCP" / Addtl-Link

   Addtl-Link      = Atom
                   ; Additional standard names for links are
                   ; registered with the Internet Assigned Numbers
                   ; Authority (IANA).  "Via" is primarily of value
                   ; with non-Internet transports.  SMTP servers
                   ; SHOULD NOT use unregistered names.

   Protocol        = "ESMTP" / "SMTP" / Attdl-Protocol

   Attdl-Protocol = Atom
                   ; Additional standard names for protocols are
                   ; registered with the Internet Assigned Numbers
                   ; Authority (IANA) in the "mail parameters"
                   ; registry [9].  SMTP servers SHOULD NOT
                   ; use unregistered names.
```

4.5. Additional Implementation Issues

4.5.1. Minimum Implementation

In order to make SMTP workable, the following minimum implementation MUST be provided by all receivers. The following commands MUST be supported to conform to this specification:

EHLO
HELO
MAIL
RCPT
DATA
RSET
NOOP
QUIT
VRFY

Any system that includes an SMTP server supporting mail relaying or delivery MUST support the reserved mailbox "postmaster" as a case-insensitive local name. This postmaster address is not strictly necessary if the server always returns 554 on connection opening (as described in Section 3.1). The requirement to accept mail for postmaster implies that RCPT commands that specify a mailbox for postmaster at any of the domains for which the SMTP server provides mail service, as well as the special case of "RCPT TO:<Postmaster>" (with no domain specification), MUST be supported.

SMTP systems are expected to make every reasonable effort to accept mail directed to Postmaster from any other system on the Internet. In extreme cases -- such as to contain a denial of service attack or other breach of security -- an SMTP server may block mail directed to Postmaster. However, such arrangements SHOULD be narrowly tailored so as to avoid blocking messages that are not part of such attacks.

4.5.2. Transparency

Without some provision for data transparency, the character sequence "<CRLF>.<CRLF>" ends the mail text and cannot be sent by the user. In general, users are not aware of such "forbidden" sequences. To allow all user composed text to be transmitted transparently, the following procedures are used:

- Before sending a line of mail text, the SMTP client checks the first character of the line. If it is a period, one additional period is inserted at the beginning of the line.

o When a line of mail text is received by the SMTP server, it checks the line. If the line is composed of a single period, it is treated as the end of mail indicator. If the first character is a period and there are other characters on the line, the first character is deleted.

The mail data may contain any of the 128 ASCII characters. All characters are to be delivered to the recipient's mailbox, including spaces, vertical and horizontal tabs, and other control characters. If the transmission channel provides an 8-bit byte (octet) data stream, the 7-bit ASCII codes are transmitted, right justified, in the octets, with the high-order bits cleared to zero. See Section 3.6 for special treatment of these conditions in SMTP systems serving a relay function.

In some systems, it may be necessary to transform the data as it is received and stored. This may be necessary for hosts that use a different character set than ASCII as their local character set, that store data in records rather than strings, or which use special character sequences as delimiters inside mailboxes. If such transformations are necessary, they MUST be reversible, especially if they are applied to mail being relayed.

4.5.3. Sizes and Timeouts

4.5.3.1. Size Limits and Minimums

There are several objects that have required minimum/maximum sizes. Every implementation MUST be able to receive objects of at least these sizes. Objects larger than these sizes SHOULD be avoided when possible. However, some Internet mail constructs such as encoded X.400 addresses (RFC 2156 [35]) will often require larger objects. Clients MAY attempt to transmit these, but MUST be prepared for a server to reject them if they cannot be handled by it. To the maximum extent possible, implementation techniques that impose no limits on the length of these objects should be used.

Extensions to SMTP may involve the use of characters that occupy more than a single octet each. This section therefore specifies lengths in octets where absolute lengths, rather than character counts, are intended.

4.5.3.1.1. Local-part

The maximum total length of a user name or other local-part is 64 octets.

4.5.3.1.2. Domain

The maximum total length of a domain name or number is 255 octets.

4.5.3.1.3. Path

The maximum total length of a reverse-path or forward-path is 256 octets (including the punctuation and element separators).

4.5.3.1.4. Command Line

The maximum total length of a command line including the command word and the <CRLF> is 512 octets. SMTP extensions may be used to increase this limit.

4.5.3.1.5. Reply Line

The maximum total length of a reply line including the reply code and the <CRLF> is 512 octets. More information may be conveyed through multiple-line replies.

4.5.3.1.6. Text Line

The maximum total length of a text line including the <CRLF> is 1000 octets (not counting the leading dot duplicated for transparency). This number may be increased by the use of SMTP Service Extensions.

4.5.3.1.7. Message Content

The maximum total length of a message content (including any message header section as well as the message body) MUST BE at least 64K octets. Since the introduction of Internet Standards for multimedia mail (RFC 2045 [21]), message lengths on the Internet have grown dramatically, and message size restrictions should be avoided if at all possible. SMTP server systems that must impose restrictions SHOULD implement the "SIZE" service extension of RFC 1870 [10], and SMTP client systems that will send large messages SHOULD utilize it when possible.

4.5.3.1.8. Recipients Buffer

The minimum total number of recipients that MUST be buffered is 100 recipients. Rejection of messages (for excessive recipients) with fewer than 100 RCPT commands is a violation of this specification. The general principle that relaying SMTP server MUST NOT, and delivery SMTP servers SHOULD NOT, perform validation tests on message header fields suggests that messages SHOULD NOT be rejected based on the total number of recipients shown in header fields. A server that imposes a limit on the number of recipients MUST behave in an orderly fashion, such as rejecting additional addresses over its limit rather than silently discarding addresses previously accepted. A client that needs to deliver a message containing over 100 RCPT commands SHOULD be prepared to transmit in 100-recipient "chunks" if the server declines to accept more than 100 recipients in a single message.

4.5.3.1.9. Treatment When Limits Exceeded

Errors due to exceeding these limits may be reported by using the reply codes. Some examples of reply codes are:

500 Line too long.

or

501 Path too long

or

452 Too many recipients (see below)

or

552 Too much mail data.

4.5.3.1.10. Too Many Recipients Code

RFC 821 [1] incorrectly listed the error where an SMTP server exhausts its implementation limit on the number of RCPT commands ("too many recipients") as having reply code 552. The correct reply code for this condition is 452. Clients SHOULD treat a 552 code in this case as a temporary, rather than permanent, failure so the logic below works.

When a conforming SMTP server encounters this condition, it has at least 100 successful RCPT commands in its recipients buffer. If the server is able to accept the message, then at least these 100 addresses will be removed from the SMTP client's queue. When the client attempts retransmission of those addresses that received 452

responses, at least 100 of these will be able to fit in the SMTP server's recipients buffer. Each retransmission attempt that is able to deliver anything will be able to dispose of at least 100 of these recipients.

If an SMTP server has an implementation limit on the number of RCPT commands and this limit is exhausted, it MUST use a response code of 452 (but the client SHOULD also be prepared for a 552, as noted above). If the server has a configured site-policy limitation on the number of RCPT commands, it MAY instead use a 5yz response code. In particular, if the intent is to prohibit messages with more than a site-specified number of recipients, rather than merely limit the number of recipients in a given mail transaction, it would be reasonable to return a 503 response to any DATA command received subsequent to the 452 (or 552) code or to simply return the 503 after DATA without returning any previous negative response.

4.5.3.2. Timeouts

An SMTP client MUST provide a timeout mechanism. It MUST use per-command timeouts rather than somehow trying to time the entire mail transaction. Timeouts SHOULD be easily reconfigurable, preferably without recompiling the SMTP code. To implement this, a timer is set for each SMTP command and for each buffer of the data transfer. The latter means that the overall timeout is inherently proportional to the size of the message.

Based on extensive experience with busy mail-relay hosts, the minimum per-command timeout values SHOULD be as follows:

4.5.3.2.1. Initial 220 Message: 5 Minutes

An SMTP client process needs to distinguish between a failed TCP connection and a delay in receiving the initial 220 greeting message. Many SMTP servers accept a TCP connection but delay delivery of the 220 message until their system load permits more mail to be processed.

4.5.3.2.2. MAIL Command: 5 Minutes

4.5.3.2.3. RCPT Command: 5 Minutes

A longer timeout is required if processing of mailing lists and aliases is not deferred until after the message was accepted.

4.5.3.2.4. DATA Initiation: 2 Minutes

This is while awaiting the "354 Start Input" reply to a DATA command.

4.5.3.2.5. Data Block: 3 Minutes

This is while awaiting the completion of each TCP SEND call transmitting a chunk of data.

4.5.3.2.6. DATA Termination: 10 Minutes.

This is while awaiting the "250 OK" reply. When the receiver gets the final period terminating the message data, it typically performs processing to deliver the message to a user mailbox. A spurious timeout at this point would be very wasteful and would typically result in delivery of multiple copies of the message, since it has been successfully sent and the server has accepted responsibility for delivery. See Section 6.1 for additional discussion.

4.5.3.2.7. Server Timeout: 5 Minutes.

An SMTP server SHOULD have a timeout of at least 5 minutes while it is awaiting the next command from the sender.

4.5.4. Retry Strategies

The common structure of a host SMTP implementation includes user mailboxes, one or more areas for queuing messages in transit, and one or more daemon processes for sending and receiving mail. The exact structure will vary depending on the needs of the users on the host and the number and size of mailing lists supported by the host. We describe several optimizations that have proved helpful, particularly for mailers supporting high traffic levels.

Any queuing strategy MUST include timeouts on all activities on a per-command basis. A queuing strategy MUST NOT send error messages in response to error messages under any circumstances.

4.5.4.1. Sending Strategy

The general model for an SMTP client is one or more processes that periodically attempt to transmit outgoing mail. In a typical system, the program that composes a message has some method for requesting immediate attention for a new piece of outgoing mail, while mail that cannot be transmitted immediately MUST be queued and periodically retried by the sender. A mail queue entry will include not only the message itself but also the envelope information.

The sender MUST delay retrying a particular destination after one attempt has failed. In general, the retry interval SHOULD be at least 30 minutes; however, more sophisticated and variable strategies will be beneficial when the SMTP client can determine the reason for non-delivery.

Retries continue until the message is transmitted or the sender gives up; the give-up time generally needs to be at least 4-5 days. It MAY be appropriate to set a shorter maximum number of retries for non-delivery notifications and equivalent error messages than for standard messages. The parameters to the retry algorithm MUST be configurable.

A client SHOULD keep a list of hosts it cannot reach and corresponding connection timeouts, rather than just retrying queued mail items.

Experience suggests that failures are typically transient (the target system or its connection has crashed), favoring a policy of two connection attempts in the first hour the message is in the queue, and then backing off to one every two or three hours.

The SMTP client can shorten the queuing delay in cooperation with the SMTP server. For example, if mail is received from a particular address, it is likely that mail queued for that host can now be sent. Application of this principle may, in many cases, eliminate the requirement for an explicit "send queues now" function such as ETRN, RFC 1985 [36].

The strategy may be further modified as a result of multiple addresses per host (see below) to optimize delivery time versus resource usage.

An SMTP client may have a large queue of messages for each unavailable destination host. If all of these messages were retried in every retry cycle, there would be excessive Internet overhead and the sending system would be blocked for a long period. Note that an SMTP client can generally determine that a delivery attempt has failed only after a timeout of several minutes, and even a one-minute timeout per connection will result in a very large delay if retries are repeated for dozens, or even hundreds, of queued messages to the same host.

At the same time, SMTP clients SHOULD use great care in caching negative responses from servers. In an extreme case, if EHLO is issued multiple times during the same SMTP connection, different answers may be returned by the server. More significantly, 5yz responses to the MAIL command MUST NOT be cached.

When a mail message is to be delivered to multiple recipients, and the SMTP server to which a copy of the message is to be sent is the same for multiple recipients, then only one copy of the message SHOULD be transmitted. That is, the SMTP client SHOULD use the command sequence: MAIL, RCPT, RCPT, ..., RCPT, DATA instead of the sequence: MAIL, RCPT, DATA, ..., MAIL, RCPT, DATA. However, if there are very many addresses, a limit on the number of RCPT commands per MAIL command MAY be imposed. This efficiency feature SHOULD be implemented.

Similarly, to achieve timely delivery, the SMTP client MAY support multiple concurrent outgoing mail transactions. However, some limit may be appropriate to protect the host from devoting all its resources to mail.

4.5.4.2. Receiving Strategy

The SMTP server SHOULD attempt to keep a pending listen on the SMTP port (specified by IANA as port 25) at all times. This requires the support of multiple incoming TCP connections for SMTP. Some limit MAY be imposed, but servers that cannot handle more than one SMTP transaction at a time are not in conformance with the intent of this specification.

As discussed above, when the SMTP server receives mail from a particular host address, it could activate its own SMTP queuing mechanisms to retry any mail pending for that host address.

4.5.5. Messages with a Null Reverse-Path

There are several types of notification messages that are required by existing and proposed Standards to be sent with a null reverse-path, namely non-delivery notifications as discussed in Section 3.7, other kinds of Delivery Status Notifications (DSNs, RFC 3461 [32]), and Message Disposition Notifications (MDNs, RFC 3798 [37]). All of these kinds of messages are notifications about a previous message, and they are sent to the reverse-path of the previous mail message. (If the delivery of such a notification message fails, that usually indicates a problem with the mail system of the host to which the notification message is addressed. For this reason, at some hosts the MTA is set up to forward such failed notification messages to someone who is able to fix problems with the mail system, e.g., via the postmaster alias.)

All other types of messages (i.e., any message which is not required by a Standards-Track RFC to have a null reverse-path) SHOULD be sent with a valid, non-null reverse-path.

Implementers of automated email processors should be careful to make sure that the various kinds of messages with a null reverse-path are handled correctly. In particular, such systems SHOULD NOT reply to messages with a null reverse-path, and they SHOULD NOT add a non-null reverse-path, or change a null reverse-path to a non-null one, to such messages when forwarding.

5. Address Resolution and Mail Handling

5.1. Locating the Target Host

Once an SMTP client lexically identifies a domain to which mail will be delivered for processing (as described in Sections 2.3.5 and 3.6), a DNS lookup MUST be performed to resolve the domain name (RFC 1035

[2]). The names are expected to be fully-qualified domain names (FQDNs): mechanisms for inferring FQDNs from partial names or local aliases are outside of this specification. Due to a history of problems, SMTP servers used for initial submission of messages SHOULD NOT make such inferences (Message Submission Servers [18] have somewhat more flexibility) and intermediate (relay) SMTP servers MUST NOT make them.

The lookup first attempts to locate an MX record associated with the name. If a CNAME record is found, the resulting name is processed as if it were the initial name. If a non-existent domain error is returned, this situation MUST be reported as an error. If a temporary error is returned, the message MUST be queued and retried later (see Section 4.5.4.1). If an empty list of MXs is returned, the address is treated as if it was associated with an implicit MX RR, with a preference of 0, pointing to that host. If MX records are present, but none of them are usable, or the implicit MX is unusable, this situation MUST be reported as an error.

If one or more MX RRs are found for a given name, SMTP systems MUST NOT utilize any address RRs associated with that name unless they are located using the MX RRs; the "implicit MX" rule above applies only if there are no MX records present. If MX records are present, but none of them are usable, this situation MUST be reported as an error.

When a domain name associated with an MX RR is looked up and the associated data field obtained, the data field of that response MUST contain a domain name. That domain name, when queried, MUST return at least one address record (e.g., A or AAAA RR) that gives the IP address of the SMTP server to which the message should be directed. Any other response, specifically including a value that will return a CNAME record when queried, lies outside the scope of this Standard. The prohibition on labels in the data that resolve to CNAMEs is discussed in more detail in RFC 2181, Section 10.3 [38].

When the lookup succeeds, the mapping can result in a list of alternative delivery addresses rather than a single address, because of multiple MX records, multihoming, or both. To provide reliable mail transmission, the SMTP client MUST be able to try (and retry) each of the relevant addresses in this list in order, until a delivery attempt succeeds. However, there MAY also be a configurable limit on the number of alternate addresses that can be tried. In any case, the SMTP client SHOULD try at least two addresses.

Two types of information are used to rank the host addresses: multiple MX records, and multihomed hosts.

MX records contain a preference indication that MUST be used in sorting if more than one such record appears (see below). Lower numbers are more preferred than higher ones. If there are multiple destinations with the same preference and there is no clear reason to favor one (e.g., by recognition of an easily reached address), then the sender-SMTP MUST randomize them to spread the load across multiple mail exchangers for a specific organization.

The destination host (perhaps taken from the preferred MX record) may be multihomed, in which case the domain name resolver will return a list of alternative IP addresses. It is the responsibility of the domain name resolver interface to have ordered this list by decreasing preference if necessary, and the SMTP sender MUST try them in the order presented.

Although the capability to try multiple alternative addresses is required, specific installations may want to limit or disable the use of alternative addresses. The question of whether a sender should attempt retries using the different addresses of a multihomed host has been controversial. The main argument for using the multiple addresses is that it maximizes the probability of timely delivery,

and indeed sometimes the probability of any delivery; the counter-argument is that it may result in unnecessary resource use. Note that resource use is also strongly determined by the sending strategy discussed in Section 4.5.4.1.

If an SMTP server receives a message with a destination for which it is a designated Mail eXchanger, it MAY relay the message (potentially after having rewritten the MAIL FROM and/or RCPT TO addresses), make final delivery of the message, or hand it off using some mechanism outside the SMTP-provided transport environment. Of course, neither of the latter require that the list of MX records be examined further.

If it determines that it should relay the message without rewriting the address, it MUST sort the MX records to determine candidates for delivery. The records are first ordered by preference, with the lowest-numbered records being most preferred. The relay host MUST then inspect the list for any of the names or addresses by which it might be known in mail transactions. If a matching record is found, all records at that preference level and higher-numbered ones MUST be discarded from consideration. If there are no records left at that point, it is an error condition, and the message MUST be returned as undeliverable. If records do remain, they SHOULD be tried, best preference first, as described above.

5.2. IPv6 and MX Records

In the contemporary Internet, SMTP clients and servers may be hosted on IPv4 systems, IPv6 systems, or dual-stack systems that are compatible with either version of the Internet Protocol. The host domains to which MX records point may, consequently, contain "A RR"s (IPv4), "AAAA RR"s (IPv6), or any combination of them. While RFC 3974 [39] discusses some operational experience in mixed environments, it was not comprehensive enough to justify standardization, and some of its recommendations appear to be inconsistent with this specification. The appropriate actions to be taken either will depend on local circumstances, such as performance of the relevant networks and any conversions that might be necessary, or will be obvious (e.g., an IPv6-only client need not attempt to look up A RRs or attempt to reach IPv4-only servers). Designers of SMTP implementations that might run in IPv6 or dual-stack environments should study the procedures above, especially the comments about multihomed hosts, and, preferably, provide mechanisms to facilitate operational tuning and mail interoperability between IPv4 and IPv6 systems while considering local circumstances.

6. Problem Detection and Handling

6.1. Reliable Delivery and Replies by Email

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. It MUST NOT lose the message for frivolous reasons, such as because the host later crashes or because of a predictable resource shortage. Some reasons that are not considered frivolous are discussed in the next subsection and in Section 7.8.

If there is a delivery failure after acceptance of a message, the receiver-SMTP MUST formulate and mail a notification message. This notification MUST be sent using a null ("<>") reverse-path in the envelope. The recipient of this notification MUST be the address from the envelope return path (or the Return-Path: line). However, if this address is null ("<>"), the receiver-SMTP MUST NOT send a notification. Obviously, nothing in this section can or should prohibit local decisions (i.e., as part of the same system environment as the receiver-SMTP) to log or otherwise transmit information about null address events locally if that is desired. If

the address is an explicit source route, it MUST be stripped down to its final hop.

For example, suppose that an error notification must be sent for a message that arrived with:

MAIL FROM:<@a,@b:user@d>

The notification message MUST be sent using:

RCPT TO:<user@d>

Some delivery failures after the message is accepted by SMTP will be unavoidable. For example, it may be impossible for the receiving SMTP server to validate all the delivery addresses in RCPT command(s) due to a "soft" domain system error, because the target is a mailing list (see earlier discussion of RCPT), or because the server is acting as a relay and has no immediate access to the delivering system.

To avoid receiving duplicate messages as the result of timeouts, a receiver-SMTP MUST seek to minimize the time required to respond to the final <CRLF>.<CRLF> end of data indicator. See RFC 1047 [40] for a discussion of this problem.

6.2. Unwanted, Unsolicited, and "Attack" Messages

Utility and predictability of the Internet mail system requires that messages that can be delivered should be delivered, regardless of any syntax or other faults associated with those messages and regardless of their content. If they cannot be delivered, and cannot be rejected by the SMTP server during the SMTP transaction, they should be "bounced" (returned with non-delivery notification messages) as described above. In today's world, in which many SMTP server operators have discovered that the quantity of undesirable bulk email vastly exceeds the quantity of desired mail and in which accepting a message may trigger additional undesirable traffic by providing verification of the address, those principles may not be practical.

As discussed in Section 7.8 and Section 7.9 below, dropping mail without notification of the sender is permitted in practice. However, it is extremely dangerous and violates a long tradition and community expectations that mail is either delivered or returned. If silent message-dropping is misused, it could easily undermine confidence in the reliability of the Internet's mail systems. So silent dropping of messages should be considered only in those cases where there is very high confidence that the messages are seriously fraudulent or otherwise inappropriate.

To stretch the principle of delivery if possible even further, it may be a rational policy to not deliver mail that has an invalid return address, although the history of the network is that users are typically better served by delivering any message that can be delivered. Reliably determining that a return address is invalid can be a difficult and time-consuming process, especially if the putative sending system is not directly accessible or does not fully and accurately support VRFY and, even if a "drop messages with invalid return addresses" policy is adopted, it SHOULD be applied only when there is near-certainty that the return addresses are, in fact, invalid.

Conversely, if a message is rejected because it is found to contain hostile content (a decision that is outside the scope of an SMTP server as defined in this document), rejection ("bounce") messages SHOULD NOT be sent unless the receiving site is confident that those messages will be usefully delivered. The preference and default in these cases is to avoid sending non-delivery messages when the incoming message is determined to contain hostile content.

6.3. Loop Detection

Simple counting of the number of "Received:" header fields in a message has proven to be an effective, although rarely optimal, method of detecting loops in mail systems. SMTP servers using this technique SHOULD use a large rejection threshold, normally at least 100 Received entries. Whatever mechanisms are used, servers MUST contain provisions for detecting and stopping trivial loops.

6.4. Compensating for Irregularities

Unfortunately, variations, creative interpretations, and outright violations of Internet mail protocols do occur; some would suggest that they occur quite frequently. The debate as to whether a well-behaved SMTP receiver or relay should reject a malformed message, attempt to pass it on unchanged, or attempt to repair it to increase the odds of successful delivery (or subsequent reply) began almost with the dawn of structured network mail and shows no signs of abating. Advocates of rejection claim that attempted repairs are rarely completely adequate and that rejection of bad messages is the only way to get the offending software repaired. Advocates of "repair" or "deliver no matter what" argue that users prefer that mail go through it if at all possible and that there are significant market pressures in that direction. In practice, these market pressures may be more important to particular vendors than strict conformance to the standards, regardless of the preference of the actual developers.

The problems associated with ill-formed messages were exacerbated by the introduction of the split-UA mail reading protocols (Post Office Protocol (POP) version 2 [15], Post Office Protocol (POP) version 3 [16], IMAP version 2 [41], and PCMAIL [42]). These protocols encouraged the use of SMTP as a posting (message submission) protocol, and SMTP servers as relay systems for these client hosts (which are often only intermittently connected to the Internet). Historically, many of those client machines lacked some of the mechanisms and information assumed by SMTP (and indeed, by the mail format protocol, RFC 822 [28]). Some could not keep adequate track of time; others had no concept of time zones; still others could not identify their own names or addresses; and, of course, none could satisfy the assumptions that underlay RFC 822's conception of authenticated addresses.

In response to these weak SMTP clients, many SMTP systems now complete messages that are delivered to them in incomplete or incorrect form. This strategy is generally considered appropriate when the server can identify or authenticate the client, and there are prior agreements between them. By contrast, there is at best great concern about fixes applied by a relay or delivery SMTP server that has little or no knowledge of the user or client machine. Many of these issues are addressed by using a separate protocol, such as that defined in RFC 4409 [18], for message submission, rather than using originating SMTP servers for that purpose.

The following changes to a message being processed MAY be applied when necessary by an originating SMTP server, or one used as the target of SMTP as an initial posting (message submission) protocol:

- Addition of a message-id field when none appears
- Addition of a date, time, or time zone when none appears
- Correction of addresses to proper FQDN format

The less information the server has about the client, the less likely these changes are to be correct and the more caution and conservatism should be applied when considering whether or not to perform fixes

and how. These changes MUST NOT be applied by an SMTP server that provides an intermediate relay function.
In all cases, properly operating clients supplying correct information are preferred to corrections by the SMTP server. In all cases, documentation SHOULD be provided in trace header fields and/or header field comments for actions performed by the servers.

7. Security Considerations

7.1. Mail Security and Spoofing

SMTP mail is inherently insecure in that it is feasible for even fairly casual users to negotiate directly with receiving and relaying SMTP servers and create messages that will trick a naive recipient into believing that they came from somewhere else. Constructing such a message so that the "spoofed" behavior cannot be detected by an expert is somewhat more difficult, but not sufficiently so as to be a deterrent to someone who is determined and knowledgeable. Consequently, as knowledge of Internet mail increases, so does the knowledge that SMTP mail inherently cannot be authenticated, or integrity checks provided, at the transport level. Real mail security lies only in end-to-end methods involving the message bodies, such as those that use digital signatures (see RFC 1847 [43] and, e.g., Pretty Good Privacy (PGP) in RFC 4880 [44] or Secure/ Multipurpose Internet Mail Extensions (S/MIME) in RFC 3851 [45]).

Various protocol extensions and configuration options that provide authentication at the transport level (e.g., from an SMTP client to an SMTP server) improve somewhat on the traditional situation described above. However, in general, they only authenticate one server to another rather than a chain of relays and servers, much less authenticating users or user machines. Consequently, unless they are accompanied by careful handoffs of responsibility in a carefully designed trust environment, they remain inherently weaker than end-to-end mechanisms that use digitally signed messages rather than depending on the integrity of the transport system.

Efforts to make it more difficult for users to set envelope return path and header "From" fields to point to valid addresses other than their own are largely misguided: they frustrate legitimate applications in which mail is sent by one user on behalf of another, in which error (or normal) replies should be directed to a special address, or in which a single message is sent to multiple recipients on different hosts. (Systems that provide convenient ways for users to alter these header fields on a per-message basis should attempt to establish a primary and permanent mailbox address for the user so that Sender header fields within the message data can be generated sensibly.)

This specification does not further address the authentication issues associated with SMTP other than to advocate that useful functionality not be disabled in the hope of providing some small margin of protection against a user who is trying to fake mail.

7.2. "Blind" Copies

Addresses that do not appear in the message header section may appear in the RCPT commands to an SMTP server for a number of reasons. The two most common involve the use of a mailing address as a "list exploder" (a single address that resolves into multiple addresses) and the appearance of "blind copies". Especially when more than one RCPT command is present, and in order to avoid defeating some of the purpose of these mechanisms, SMTP clients and servers SHOULD NOT copy the full set of RCPT command arguments into the header section, either as part of trace header fields or as informational or private-extension header fields. Since this rule is often violated in practice, and cannot be enforced, sending SMTP systems that are aware of "bcc" use MAY find it helpful to send each blind copy as a

separate message transaction containing only a single RCPT command.

There is no inherent relationship between either "reverse" (from MAIL, SAML, etc., commands) or "forward" (RCPT) addresses in the SMTP transaction ("envelope") and the addresses in the header section. Receiving systems SHOULD NOT attempt to deduce such relationships and use them to alter the header section of the message for delivery. The popular "Apparently-to" header field is a violation of this principle as well as a common source of unintended information disclosure and SHOULD NOT be used.

7.3. VRFY, EXPN, and Security

As discussed in Section 3.5, individual sites may want to disable either or both of VRFY or EXPN for security reasons (see below). As a corollary to the above, implementations that permit this MUST NOT appear to have verified addresses that are not, in fact, verified. If a site disables these commands for security reasons, the SMTP server MUST return a 252 response, rather than a code that could be confused with successful or unsuccessful verification.

Returning a 250 reply code with the address listed in the VRFY command after having checked it only for syntax violates this rule. Of course, an implementation that "supports" VRFY by always returning 550 whether or not the address is valid is equally not in conformance.

On the public Internet, the contents of mailing lists have become popular as an address information source for so-called "spammers."

The use of EXPN to "harvest" addresses has increased as list administrators have installed protections against inappropriate uses of the lists themselves. However, VRFY and EXPN are still useful for authenticated users and within an administrative domain. For example, VRFY and EXPN are useful for performing internal audits of how email gets routed to check and to make sure no one is automatically forwarding sensitive mail outside the organization. Sites implementing SMTP authentication may choose to make VRFY and EXPN available only to authenticated requestors. Implementations SHOULD still provide support for EXPN, but sites SHOULD carefully evaluate the tradeoffs.

Whether disabling VRFY provides any real marginal security depends on a series of other conditions. In many cases, RCPT commands can be used to obtain the same information about address validity. On the other hand, especially in situations where determination of address validity for RCPT commands is deferred until after the DATA command is received, RCPT may return no information at all, while VRFY is expected to make a serious attempt to determine validity before generating a response code (see discussion above).

7.4. Mail Rerouting Based on the 251 and 551 Response Codes

Before a client uses the 251 or 551 reply codes from a RCPT command to automatically update its future behavior (e.g., updating the user's address book), it should be certain of the server's authenticity. If it does not, it may be subject to a man in the middle attack.

7.5. Information Disclosure in Announcements

There has been an ongoing debate about the tradeoffs between the debugging advantages of announcing server type and version (and, sometimes, even server domain name) in the greeting response or in response to the HELP command and the disadvantages of exposing information that might be useful in a potential hostile attack. The utility of the debugging information is beyond doubt. Those who argue for making it available point out that it is far better to

actually secure an SMTP server rather than hope that trying to conceal known vulnerabilities by hiding the server's precise identity will provide more protection. Sites are encouraged to evaluate the tradeoff with that issue in mind; implementations SHOULD minimally provide for making type and version information available in some way to other network hosts.

7.6. Information Disclosure in Trace Fields

In some circumstances, such as when mail originates from within a LAN whose hosts are not directly on the public Internet, trace ("Received") header fields produced in conformance with this specification may disclose host names and similar information that would not normally be available. This ordinarily does not pose a problem, but sites with special concerns about name disclosure should be aware of it. Also, the optional FOR clause should be supplied with caution or not at all when multiple recipients are involved lest it inadvertently disclose the identities of "blind copy" recipients to others.

7.7. Information Disclosure in Message Forwarding

As discussed in Section 3.4, use of the 251 or 551 reply codes to identify the replacement address associated with a mailbox may inadvertently disclose sensitive information. Sites that are concerned about those issues should ensure that they select and configure servers appropriately.

7.8. Resistance to Attacks

In recent years, there has been an increase of attacks on SMTP servers, either in conjunction with attempts to discover addresses for sending unsolicited messages or simply to make the servers inaccessible to others (i.e., as an application-level denial of service attack). While the means of doing so are beyond the scope of this Standard, rational operational behavior requires that servers be permitted to detect such attacks and take action to defend themselves. For example, if a server determines that a large number of RCPT TO commands are being sent, most or all with invalid addresses, as part of such an attack, it would be reasonable for the server to close the connection after generating an appropriate number of 5yz (normally 550) replies.

7.9. Scope of Operation of SMTP Servers

It is a well-established principle that an SMTP server may refuse to accept mail for any operational or technical reason that makes sense to the site providing the server. However, cooperation among sites and installations makes the Internet possible. If sites take excessive advantage of the right to reject traffic, the ubiquity of email availability (one of the strengths of the Internet) will be threatened; considerable care should be taken and balance maintained if a site decides to be selective about the traffic it will accept and process.

In recent years, use of the relay function through arbitrary sites has been used as part of hostile efforts to hide the actual origins of mail. Some sites have decided to limit the use of the relay function to known or identifiable sources, and implementations SHOULD provide the capability to perform this type of filtering. When mail is rejected for these or other policy reasons, a 550 code SHOULD be used in response to EHLO (or HELO), MAIL, or RCPT as appropriate.

8. IANA Considerations

IANA maintains three registries in support of this specification, all of which were created for RFC 2821 or earlier. This document expands the third one as specified below. The registry references listed are

as of the time of publication; IANA does not guarantee the locations associated with the URLs. The registries are as follows:

- The first, "Simple Mail Transfer Protocol (SMTP) Service Extensions" [46], consists of SMTP service extensions with the associated keywords, and, as needed, parameters and verbs. As specified in Section 2.2.2, no entry may be made in this registry that starts in an "X". Entries may be made only for service extensions (and associated keywords, parameters, or verbs) that are defined in Standards-Track or Experimental RFCs specifically approved by the IESG for this purpose.

- The second registry, "Address Literal Tags" [47], consists of "tags" that identify forms of domain literals other than those for IPv4 addresses (specified in RFC 821 and in this document). The initial entry in that registry is for IPv6 addresses (specified in this document). Additional literal types require standardization before being used; none are anticipated at this time.

- The third, "Mail Transmission Types" [46], established by RFC 821 and renewed by this specification, is a registry of link and protocol identifiers to be used with the "via" and "with" subclauses of the time stamp ("Received:" header field) described in Section 4.4. Link and protocol identifiers in addition to those specified in this document may be registered only by standardization or by way of an RFC-documented, IESG-approved, Experimental protocol extension. This name space is for identification and not limited in size: the IESG is encouraged to approve on the basis of clear documentation and a distinct method rather than preferences about the properties of the method itself.

  An additional subsection has been added to the "VIA link types" and "WITH protocol types" subsections of this registry to contain registrations of "Additional-registered-clauses" as described above. The registry will contain clause names, a description, a summary of the syntax of the associated String, and a reference. As new clauses are defined, they may, in principle, specify creation of their own registries if the Strings consist of reserved terms or keywords rather than less restricted strings. As with link and protocol identifiers, additional clauses may be registered only by standardization or by way of an RFC-documented, IESG-approved, Experimental protocol extension. The additional clause name space is for identification and is not limited in size: the IESG is encouraged to approve on the basis of clear documentation, actual use or strong signs that the clause will be used, and a distinct requirement rather than preferences about the properties of the clause itself.

In addition, if additional trace header fields (i.e., in addition to Return-path and Received) are ever created, those trace fields MUST be added to the IANA registry established by BCP 90 (RFC 3864) [11] for use with RFC 5322 [4].

9. Acknowledgments

Many people contributed to the development of RFC 2821. That document should be consulted for those acknowledgments. For the present document, the editor and the community owe thanks to Dawn Mann and Tony Hansen who assisted in the very painful process of editing and converting the internal format of the document from one system to another.

Neither this document nor RFC 2821 would have been possible without the many contribution and insights of the late Jon Postel. Those contributions of course include the original specification of SMTP in RFC 821. A considerable quantity of text from RFC 821 still appears in this document as do several of Jon's original examples that have been updated only as needed to reflect other changes in the

specification.

Many people made comments or suggestions on the mailing list or in notes to the author. Important corrections or clarifications were suggested by several people, including Matti Aarnio, Glenn Anderson, Derek J. Balling, Alex van den Bogaerdt, Stephane Bortzmeyer, Vint Cerf, Jutta Degener, Steve Dorner, Lisa Dusseault, Frank Ellerman, Ned Freed, Randy Gellens, Sabahattin Gucukoglu, Philip Guenther, Arnt Gulbrandsen, Eric Hall, Richard O. Hammer, Tony Hansen, Peter J. Holzer, Kari Hurtta, Bryon Roche Kain, Valdis Kletnieks, Mathias Koerber, John Leslie, Bruce Lilly, Jeff Macdonald, Mark E. Mallett, Mark Martinec, S. Moonesamy, Lyndon Nerenberg, Chris Newman, Douglas Otis, Pete Resnick, Robert A. Rosenberg, Vince Sabio, Hector Santos, David F. Skoll, Paul Smith, and Brett Watson.

The efforts of the Area Directors -- Lisa Dusseault, Ted Hardie, and Chris Newman -- to get this effort restarted and keep it moving, and of an ad hoc committee with the same purpose, are gratefully acknowledged. The members of that committee were (in alphabetical order) Dave Crocker, Cyrus Daboo, Tony Finch, Ned Freed, Randall Gellens, Tony Hansen, the author, and Alexey Melnikov. Tony Hansen also acted as ad hoc chair on the mailing list reviewing this document; without his efforts, sense of balance and fairness, and patience, it clearly would not have been possible.

10. References

10.1. Normative References

[1] Postel, J., "Simple Mail Transfer Protocol", STD 10, RFC 821, August 1982.

[2] Mockapetris, P., "Domain names - implementation and specification", STD 13, RFC 1035, November 1987.

[3] Braden, R., "Requirements for Internet Hosts - Application and Support", STD 3, RFC 1123, October 1989.

[4] Resnick, P., "Internet Message Format", RFC 5322, October 2008.

[5] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[6] American National Standards Institute (formerly United States of America Standards Institute), "USA Code for Information Interchange", ANSI X3.4-1968, 1968.

ANSI X3.4-1968 has been replaced by newer versions with slight modifications, but the 1968 version remains definitive for the Internet.

[7] Crocker, D. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, January 2008.

[8] Hinden, R. and S. Deering, "IP Version 6 Addressing Architecture", RFC 4291, February 2006.

[9] Newman, C., "ESMTP and LMTP Transmission Types Registration", RFC 3848, July 2004.

[10] Klensin, J., Freed, N., and K. Moore, "SMTP Service Extension for Message Size Declaration", STD 10, RFC 1870, November 1995.

[11] Klyne, G., Nottingham, M., and J. Mogul, "Registration Procedures for Message Header Fields", BCP 90, RFC 3864, September 2004.

10.2. Informative References

[12] Partridge, C., "Mail routing and the domain system", RFC 974, January 1986.

[13] Klensin, J., Freed, N., Rose, M., Stefferud, E., and D. Crocker, "SMTP Service Extensions", STD 10, RFC 1869, November 1995.

[14] Klensin, J., "Simple Mail Transfer Protocol", RFC 2821, April 2001.

[15] Butler, M., Postel, J., Chase, D., Goldberger, J., and J. Reynolds, "Post Office Protocol: Version 2", RFC 937, February 1985.

[16] Myers, J. and M. Rose, "Post Office Protocol - Version 3", STD 53, RFC 1939, May 1996.

[17] Crispin, M., "INTERNET MESSAGE ACCESS PROTOCOL - VERSION 4rev1", RFC 3501, March 2003.

[18] Gellens, R. and J. Klensin, "Message Submission for Mail", RFC 4409, April 2006.

[19] Freed, N., "SMTP Service Extension for Command Pipelining", STD 60, RFC 2920, September 2000.

[20] Vaudreuil, G., "SMTP Service Extensions for Transmission of Large and Binary MIME Messages", RFC 3030, December 2000.

[21] Freed, N. and N. Borenstein, "Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies", RFC 2045, November 1996.

[22] Klensin, J., Freed, N., Rose, M., Stefferud, E., and D. Crocker, "SMTP Service Extension for 8bit-MIMEtransport", RFC 1652, July 1994.

[23] Moore, K., "MIME (Multipurpose Internet Mail Extensions) Part Three: Message Header Extensions for Non-ASCII Text", RFC 2047, November 1996.

[24] Freed, N. and K. Moore, "MIME Parameter Value and Encoded Word Extensions: Character Sets, Languages, and Continuations", RFC 2231, November 1997.

[25] Vaudreuil, G., "Enhanced Mail System Status Codes", RFC 3463, January 2003.

[26] Hansen, T. and J. Klensin, "A Registry for SMTP Enhanced Mail System Status Codes", BCP 138, RFC 5248, June 2008.

[27] Freed, N., "Behavior of and Requirements for Internet Firewalls", RFC 2979, October 2000.

[28] Crocker, D., "Standard for the format of ARPA Internet text messages", STD 11, RFC 822, August 1982.

[29] Wong, M. and W. Schlitt, "Sender Policy Framework (SPF) for Authorizing Use of Domains in E-Mail, Version 1", RFC 4408, April 2006.

[30] Fenton, J., "Analysis of Threats Motivating DomainKeys Identified Mail (DKIM)", RFC 4686, September 2006.

[31] Allman, E., Callas, J., Delany, M., Libbey, M., Fenton, J., and M. Thomas, "DomainKeys Identified Mail (DKIM) Signatures", RFC 4871, May 2007.

[32] Moore, K., "Simple Mail Transfer Protocol (SMTP) Service Extension for Delivery Status Notifications (DSNs)", RFC 3461, January 2003.

[33] Moore, K. and G. Vaudreuil, "An Extensible Message Format for Delivery Status Notifications", RFC 3464, January 2003.

[34] Postel, J. and J. Reynolds, "File Transfer Protocol", STD 9, RFC 959, October 1985.

[35] Kille, S., "MIXER (Mime Internet X.400 Enhanced Relay): Mapping between X.400 and RFC 822/MIME", RFC 2156, January 1998.

[36] De Winter, J., "SMTP Service Extension for Remote Message Queue Starting", RFC 1985, August 1996.

[37] Hansen, T. and G. Vaudreuil, "Message Disposition Notification", RFC 3798, May 2004.

[38] Elz, R. and R. Bush, "Clarifications to the DNS Specification", RFC 2181, July 1997.

[39] Nakamura, M. and J. Hagino, "SMTP Operational Experience in Mixed IPv4/v6 Environments", RFC 3974, January 2005.

[40] Partridge, C., "Duplicate messages and SMTP", RFC 1047, February 1988.

[41] Crispin, M., "Interactive Mail Access Protocol: Version 2", RFC 1176, August 1990.

[42] Lambert, M., "PCMAIL: A distributed mail system for personal computers", RFC 1056, June 1988.

[43] Galvin, J., Murphy, S., Crocker, S., and N. Freed, "Security Multiparts for MIME: Multipart/Signed and Multipart/Encrypted", RFC 1847, October 1995.

[44] Callas, J., Donnerhacke, L., Finney, H., Shaw, D., and R. Thayer, "OpenPGP Message Format", RFC 4880, November 2007.

[45] Ramsdell, B., "Secure/Multipurpose Internet Mail Extensions (S/MIME) Version 3.1 Message Specification", RFC 3851, July 2004.

[46] Internet Assigned Number Authority (IANA), "IANA Mail Parameters", 2007, <http://www.iana.org/assignments/mail-parameters>.

[47] Internet Assigned Number Authority (IANA), "Address Literal Tags", 2007, <http://www.iana.org/assignments/address-literal-tags>.

Appendix A. TCP Transport Service

The TCP connection supports the transmission of 8-bit bytes. The SMTP data is 7-bit ASCII characters. Each character is transmitted as an 8-bit byte with the high-order bit cleared to zero. Service extensions may modify this rule to permit transmission of full 8-bit data bytes as part of the message body, or, if specifically designed to do so, in SMTP commands or responses.

Appendix B. Generating SMTP Commands from RFC 822 Header Fields

Some systems use an RFC 822 header section (only) in a mail submission protocol, or otherwise generate SMTP commands from RFC 822 header fields when such a message is handed to an MTA from a UA.

While the MTA-UA protocol is a private matter, not covered by any Internet Standard, there are problems with this approach. For example, there have been repeated problems with proper handling of "bcc" copies and redistribution lists when information that conceptually belongs to the mail envelope is not separated early in processing from header field information (and kept separate).

It is recommended that the UA provide its initial ("submission client") MTA with an envelope separate from the message itself. However, if the envelope is not supplied, SMTP commands SHOULD be generated as follows:

1. Each recipient address from a TO, CC, or BCC header field SHOULD be copied to a RCPT command (generating multiple message copies if that is required for queuing or delivery). This includes any addresses listed in a RFC 822 "group". Any BCC header fields SHOULD then be removed from the header section. Once this process is completed, the remaining header fields SHOULD be checked to verify that at least one TO, CC, or BCC header field remains. If none do, then a BCC header field with no additional information SHOULD be inserted as specified in [4].

2. The return address in the MAIL command SHOULD, if possible, be derived from the system's identity for the submitting (local) user, and the "From:" header field otherwise. If there is a system identity available, it SHOULD also be copied to the Sender header field if it is different from the address in the From header field. (Any Sender header field that was already there SHOULD be removed.) Systems may provide a way for submitters to override the envelope return address, but may want to restrict its use to privileged users. This will not prevent mail forgery, but may lessen its incidence; see Section 7.1.

When an MTA is being used in this way, it bears responsibility for ensuring that the message being transmitted is valid. The mechanisms for checking that validity, and for handling (or returning) messages that are not valid at the time of arrival, are part of the MUA-MTA interface and not covered by this specification.

A submission protocol based on Standard RFC 822 information alone MUST NOT be used to gateway a message from a foreign (non-SMTP) mail system into an SMTP environment. Additional information to construct an envelope must come from some source in the other environment, whether supplemental header fields or the foreign system's envelope.

Attempts to gateway messages using only their header "To" and "Cc" fields have repeatedly caused mail loops and other behavior adverse to the proper functioning of the Internet mail environment. These problems have been especially common when the message originates from an Internet mailing list and is distributed into the foreign environment using envelope information. When these messages are then processed by a header-section-only remailer, loops back to the Internet environment (and the mailing list) are almost inevitable.

Appendix C. Source Routes

Historically, the <reverse-path> was a reverse source routing list of hosts and a source mailbox. The first host in the <reverse-path> was historically the host sending the MAIL command; today, source routes SHOULD NOT appear in the reverse-path. Similarly, the <forward-path> may be a source routing lists of hosts and a destination mailbox. However, in general, the <forward-path> SHOULD contain only a mailbox and domain name, relying on the domain name system to supply routing information if required. The use of source routes is deprecated (see Appendix F.2); while servers MUST be prepared to receive and handle them as discussed in Section 3.3 and Appendix F.2, clients SHOULD NOT transmit them and this section is included in the current specification only to provide context. It has been modified somewhat

from the material in RFC 821 to prevent server actions that might confuse clients or subsequent servers that do not expect a full source route implementation.

For relay purposes, the forward-path may be a source route of the form "@ONE,@TWO:JOE@THREE", where ONE, TWO, and THREE MUST be fully-qualified domain names. This form is used to emphasize the distinction between an address and a route. The mailbox (here, JOE@THREE) is an absolute address, and the route is information about how to get there. The two concepts should not be confused.

If source routes are used, RFC 821 and the text below should be consulted for the mechanisms for constructing and updating the forward-path. A server that is reached by means of a source route (e.g., its domain name appears first in the list in the forward-path) MUST remove its domain name from any forward-paths in which that domain name appears before forwarding the message and MAY remove all other source routing information. The reverse-path SHOULD NOT be updated by servers conforming to this specification.

Notice that the forward-path and reverse-path appear in the SMTP commands and replies, but not necessarily in the message. That is, there is no need for these paths and especially this syntax to appear in the "To:" , "From:", "CC:", etc. fields of the message header section. Conversely, SMTP servers MUST NOT derive final message routing information from message header fields.

When the list of hosts is present despite the recommendations above, it is a "reverse" source route and indicates that the mail was relayed through each host on the list (the first host in the list was the most recent relay). This list is used as a source route to return non-delivery notices to the sender. If, contrary to the recommendations here, a relay host adds itself to the beginning of the list, it MUST use its name as known in the transport environment to which it is relaying the mail rather than that of the transport environment from which the mail came (if they are different). Note that a situation could easily arise in which some relay hosts add their names to the reverse source route and others do not, generating discontinuities in the routing list. This is another reason why servers needing to return a message SHOULD ignore the source route entirely and simply use the domain as specified in the Mailbox.

Appendix D. Scenarios

This section presents complete scenarios of several types of SMTP sessions. In the examples, "C:" indicates what is said by the SMTP client, and "S:" indicates what is said by the SMTP server.

D.1. A Typical SMTP Transaction Scenario

This SMTP example shows mail sent by Smith at host bar.com, and to Jones, Green, and Brown at host foo.com. Here we assume that host bar.com contacts host foo.com directly. The mail is accepted for Jones and Brown. Green does not have a mailbox at host foo.com.

```
S: 220 foo.com Simple Mail Transfer Service Ready
C: EHLO bar.com
S: 250-foo.com greets bar.com
S: 250-8BITMIME
S: 250-SIZE
S: 250-DSN
S: 250 HELP
C: MAIL FROM:<Smith@bar.com>
S: 250 OK
C: RCPT TO:<Jones@foo.com>
S: 250 OK
C: RCPT TO:<Green@foo.com>
S: 550 No such user here
```

```
      C: RCPT TO:<Brown@foo.com>
      S: 250 OK
      C: DATA
      S: 354 Start mail input; end with <CRLF>.<CRLF>
      C: Blah blah blah...
      C: ...etc. etc. etc.
      C: .
      S: 250 OK
      C: QUIT
      S: 221 foo.com Service closing transmission channel
```

D.2. Aborted SMTP Transaction Scenario

```
      S: 220 foo.com Simple Mail Transfer Service Ready
      C: EHLO bar.com
      S: 250-foo.com greets bar.com
      S: 250-8BITMIME
      S: 250-SIZE
      S: 250-DSN
      S: 250 HELP
      C: MAIL FROM:<Smith@bar.com>
      S: 250 OK
      C: RCPT TO:<Jones@foo.com>
      S: 250 OK
      C: RCPT TO:<Green@foo.com>
      S: 550 No such user here
      C: RSET
      S: 250 OK
      C: QUIT
      S: 221 foo.com Service closing transmission channel
```

D.3. Relayed Mail Scenario

```
   Step 1 -- Source Host to Relay Host

   The source host performs a DNS lookup on XYZ.COM (the destination
   address) and finds DNS MX records specifying xyz.com as the best
   preference and foo.com as a lower preference.  It attempts to open a
   connection to xyz.com and fails.  It then opens a connection to
   foo.com, with the following dialogue:

      S: 220 foo.com Simple Mail Transfer Service Ready
      C: EHLO bar.com
      S: 250-foo.com greets bar.com
      S: 250-8BITMIME
      S: 250-SIZE
      S: 250-DSN
      S: 250 HELP
      C: MAIL FROM:<JQP@bar.com>
      S: 250 OK
      C: RCPT TO:<Jones@XYZ.COM>
      S: 250 OK
      C: DATA
      S: 354 Start mail input; end with <CRLF>.<CRLF>
      C: Date: Thu, 21 May 1998 05:33:29 -0700
      C: From: John Q. Public <JQP@bar.com>
      C: Subject: The Next Meeting of the Board
      C: To: Jones@xyz.com
      C:
      C: Bill:
      C: The next meeting of the board of directors will be
      C: on Tuesday.
      C: John.
      C: .
      S: 250 OK
      C: QUIT
      S: 221 foo.com Service closing transmission channel
```

```
   Step 2 -- Relay Host to Destination Host

   foo.com, having received the message, now does a DNS lookup on
   xyz.com.  It finds the same set of MX records, but cannot use the one
   that points to itself (or to any other host as a worse preference).
   It tries to open a connection to xyz.com itself and succeeds.  Then
   we have:

           S: 220 xyz.com Simple Mail Transfer Service Ready
           C: EHLO foo.com
           S: 250 xyz.com is on the air
           C: MAIL FROM:<JQP@bar.com>
           S: 250 OK
           C: RCPT TO:<Jones@XYZ.COM>
           S: 250 OK
           C: DATA
           S: 354 Start mail input; end with <CRLF>.<CRLF>
           C: Received: from bar.com by foo.com ; Thu, 21 May 1998
           C:     05:33:29 -0700
           C: Date: Thu, 21 May 1998 05:33:22 -0700
           C: From: John Q. Public <JQP@bar.com>
           C: Subject:  The Next Meeting of the Board
           C: To: Jones@xyz.com
           C:
           C: Bill:
           C: The next meeting of the board of directors will be
           C: on Tuesday.
           C:                              John.
           C: .
           S: 250 OK
           C: QUIT
           S: 221 foo.com Service closing transmission channel

D.4.  Verifying and Sending Scenario

       S: 220 foo.com Simple Mail Transfer Service Ready
       C: EHLO bar.com
       S: 250-foo.com greets bar.com
       S: 250-8BITMIME
       S: 250-SIZE
       S: 250-DSN
       S: 250-VRFY
       S: 250 HELP
       C: VRFY Crispin
       S: 250 Mark Crispin <Admin.MRC@foo.com>
       C: MAIL FROM:<EAK@bar.com>
       S: 250 OK
       C: RCPT TO:<Admin.MRC@foo.com>
       S: 250 OK
       C: DATA
       S: 354 Start mail input; end with <CRLF>.<CRLF>
       C: Blah blah blah...
       C: ...etc. etc. etc.
       C: .
       S: 250 OK
       C: QUIT
       S: 221 foo.com Service closing transmission channel

Appendix E.  Other Gateway Issues

   In general, gateways between the Internet and other mail systems
   SHOULD attempt to preserve any layering semantics across the
   boundaries between the two mail systems involved.  Gateway-
   translation approaches that attempt to take shortcuts by mapping
   (such as mapping envelope information from one system to the message
   header section or body of another) have generally proven to be
   inadequate in important ways.  Systems translating between
   environments that do not support both envelopes and a header section
```

and Internet mail must be written with the understanding that some information loss is almost inevitable.

Appendix F. Deprecated Features of RFC 821

A few features of RFC 821 have proven to be problematic and SHOULD NOT be used in Internet mail.

F.1. TURN

This command, described in RFC 821, raises important security issues since, in the absence of strong authentication of the host requesting that the client and server switch roles, it can easily be used to divert mail from its correct destination. Its use is deprecated; SMTP systems SHOULD NOT use it unless the server can authenticate the client.

F.2. Source Routing

RFC 821 utilized the concept of explicit source routing to get mail from one host to another via a series of relays. The requirement to utilize source routes in regular mail traffic was eliminated by the introduction of the domain name system "MX" record and the last significant justification for them was eliminated by the introduction, in RFC 1123, of a clear requirement that addresses following an "@" must all be fully-qualified domain names. Consequently, the only remaining justifications for the use of source routes are support for very old SMTP clients or MUAs and in mail system debugging. They can, however, still be useful in the latter circumstance and for routing mail around serious, but temporary, problems such as problems with the relevant DNS records.

SMTP servers MUST continue to accept source route syntax as specified in the main body of this document and in RFC 1123. They MAY, if necessary, ignore the routes and utilize only the target domain in the address. If they do utilize the source route, the message MUST be sent to the first domain shown in the address. In particular, a server MUST NOT guess at shortcuts within the source route.

Clients SHOULD NOT utilize explicit source routing except under unusual circumstances, such as debugging or potentially relaying around firewall or mail system configuration errors.

F.3. HELO

As discussed in Sections 3.1 and 4.1.1, EHLO SHOULD be used rather than HELO when the server will accept the former. Servers MUST continue to accept and process HELO in order to support older clients.

F.4. #-literals

RFC 821 provided for specifying an Internet address as a decimal integer host number prefixed by a pound sign, "#". In practice, that form has been obsolete since the introduction of TCP/IP. It is deprecated and MUST NOT be used.

F.5. Dates and Years

When dates are inserted into messages by SMTP clients or servers (e.g., in trace header fields), four-digit years MUST BE used. Two-digit years are deprecated; three-digit years were never permitted in the Internet mail system.

F.6. Sending versus Mailing

In addition to specifying a mechanism for delivering messages to user's mailboxes, RFC 821 provided additional, optional, commands to

deliver messages directly to the user's terminal screen. These commands (SEND, SAML, SOML) were rarely implemented, and changes in workstation technology and the introduction of other protocols may have rendered them obsolete even where they are implemented.

Clients SHOULD NOT provide SEND, SAML, or SOML as services. Servers MAY implement them. If they are implemented by servers, the implementation model specified in RFC 821 MUST be used and the command names MUST be published in the response to the EHLO command.

Author's Address

John C. Klensin
1770 Massachusetts Ave, Suite 322
Cambridge, MA 02140
USA

EMail: john+smtp@jck.com

Full Copyright Statement

Intellectual Property

Annexe 2

Liste des demandes de RFC sur la confiance

DNS-based Authentication of Named Entities (dane)

"Using Secure DNS to Associate Certificates with Domain Names For TLS", Paul Hoffman, Jakob Schlyter, 25-Jul-11, <draft-ietf-dane-protocol-09.txt>

"Use Cases and Requirements for DNS-based Authentication of Named Entities (DANE)", Richard Barnes, 28-Jul-11, <draft-ietf-dane-use-cases-05.txt>

Domain Keys Identified Mail (dkim)

"DKIM And Mailing Lists", Murray Kucherawy, 23-Jun-11, <draft-ietf-dkim-mailinglists-12.txt>

"DomainKeys Identified Mail (DKIM) Signatures", D. Crocker, Tony Hansen, M. Kucherawy, 11-Jul-11, <draft-ietf-dkim-rfc4871bis-15.txt>

"RFC4871 Implementation Report", Murray Kucherawy, 28-Mar-11, <draft-ietf-dkim-implementation-report-06.txt>

Host Identity Protocol (hip)

"Host Identity Protocol (HIP) Registration Extension", Julien Laganier, Teemu Koponen, Lars Eggert, 14-Mar-11, <draft-ietf-hip-rfc5203-bis-01.txt>

"Host Identity Protocol Architecture", Robert Moskowitz, 25-Feb-11, <draft-ietf-hip-rfc4423-bis-02.txt>

"An IPv6 Prefix for Overlay Routable Cryptographic Hash Identifiers (ORCHID)", Julien Laganier, Francis Dupont, 14-Mar-11, <draft-ietf-hip-rfc4843-bis-01.txt>

"Host Identity Protocol (HIP) Rendezvous Extension", Julien Laganier, Lars Eggert, 14-Mar-11, <draft-ietf-hip-rfc5204-bis-01.txt>

"Host Identity Protocol (HIP) Domain Name System (DNS) Extension", Julien Laganier, 14-Mar-11, <draft-ietf-hip-rfc5205-bis-01.txt>

"Host Identity Protocol Version 2 (HIPv2)", Robert Moskowitz, Tobias Heer, Petri Jokela, Tom Henderson, 9-Jul-11, <draft-ietf-hip-rfc5201-bis-06.txt>

"Host Mobility with the Host Identity Protocol", Pekka Nikander, Tom Henderson, Christian Vogt, Jari Arkko, 14-Mar-11, <draft-ietf-hip-rfc5206-bis-02.txt>

Messaging Abuse Reporting Format (marf)

"Extensions to DKIM for Failure Reporting", Murray Kucherawy, 15-May-11, <draft-ietf-marf-dkim-reporting-02.txt>

"A DNS TXT Record for Advertising and Discovering Willingness to Provide or Receive ARF Reports", J.D. Falk, 27-Jul-11, <draft-ietf-marf-reporting-discovery-01.txt>

"Redaction of Potentially Sensitive Data from Mail Abuse Reports", J.D. Falk, 14-Apr-11, <draft-ietf-marf-redaction-00.txt>

"Authentication Failure Reporting using the Abuse Report Format", Hilda Fontana, 28-Jun-11, <draft-ietf-marf-authfailure-report-00.txt,.pdf>

"SPF Authentication Failure Reporting using the Abuse Report Format", Scott Kitterman, 11-Jul-11, <draft-ietf-marf-spf-reporting-01.txt>

"Email Feedback Report Type Value : not-spam", Kepeng Li, Barry Leiba, 2-Jul-11, <draft-ietf-marf-not-spam-feedback-00.txt>

Individual submissions

"Overview of Email DNSBL Best Practise", Chris Lewis, Matt Sergeant, 25-Jul-11, <draft-irtf-asrg-bcp-blacklists-10.txt>

"Privacy Preferences for E-Mail Messages", Ulrich Koenig, Jan Schallaboeck, 3-Jun-11, <draft-koenig-privicons-02.txt,.ps,.pdf>

"Simple Mail Transfer Protocol extension for Alternate Recipient Delivery Option", Alexey Melnikov, 6-Apr-11, <draft-melnikov-smtp-altrecip-on-error-01.txt>

"Email Policy Service Trust Processing", Jim Schaad, 26-Jul-11, <draft-schaad-eps-trust-01.txt>

"Simple Mail Transfer Protocol extension for Message Priorities", Alexey Melnikov, Ken Carlberg, 11-Jul-11, <draft-melnikov-smtp-priority-02.txt>

"Redaction of Potentially Sensitive Data from Mail Abuse Reports", J.D. Falk, 7-Mar-11, <draft-jdfalk-marf-redaction-00.txt>

"A Reputation Vocabulary for Email Properties", Nathaniel Borenstein, Murray Kucherawy, 1-Jun-11, <draft-kucherawy-reputation-vocab-email-00.txt>

"A Reputation Vocabulary for Email Identities", Nathaniel Borenstein, Murray Kucherawy, 1-Jun-11, <draft-kucherawy-reputation-vocab-identity-00.txt>

"Updated TLS Server Identity Check Procedure for Email Related Protocols", Alexey Melnikov, 15-Jun-11, <draft-melnikov-email-tls-certs-00.txt>

"Security Labels in Internet Email", Alexey Melnikov, Kurt Zeilenga, 8-Aug-11, <draft-zeilenga-email-seclabel-01.txt>

"Internationalized Email Addresses in X.509 certificates", Alexey Melnikov, 7-Mar-11, <draft-ietf-pkix-eai-addresses-00.txt>

"Suggested values for SMTP Enhanced Status Codes for Anti-Spam Policy",

Jeff Macdonald, 5-May-11, <draft-macdonald-antispam-registry-02.txt>

"DKIM Authorized Third-Party Signers", Murray Kucherawy, 2-Aug-11, <draft-kucherawy-dkim-atps-06.txt>

"The Post Office Protocol (POP3) LIST+ Extension", Steffen Lehmann, 5-Aug-11, <draft-lehmann-morg-pop3listplus-01.txt,.pdf>

Yet Another Mail (yam)

"Preliminary Evaluation of RFC5321, Simple Mail Transfer Protocol (SMTP), for advancement from Draft Standard to Full Standard by the YAM Working Group", John Klensin, Barry Leiba, 4-May-10, <draft-ietf-yam-5321bis-smtp-pre-evaluation-05.txt>

"Message Submission for Mail", Randall Gellens, John Klensin, 27-Jul-11, <draft-ietf-yam-rfc4409bis-02.txt>

Public-Key Infrastructure (X.509) (pkix)

"Internet X.509 Public Key Infrastructure -- HTTP Transport for CMP", Tomi Kause, Martin Peylo, 30-Jun-11, <draft-ietf-pkix-cmp-transport-protocols-13.txt>

"X.509 Internet Public Key Infrastructure Online Certificate Status Protocol - OCSP", Dave Cooper, Stefan Santesson, 5-Apr-11, <draft-ietf-pkix-rfc2560bis-03.txt>

"Certificate Management over CMS (CMC) Updates", Jim Schaad, 24-Jul-11, <draft-ietf-pkix-rfc5272-bis-04.txt>

"Clarifications to the Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", Dave Cooper, 28-Mar-11, <draft-ietf-pkix-rfc5280-clarifications-02.txt>

"S/MIME Capabilities for Public Key Definitions", Jim Schaad, 6-Apr-11,

<draft-ietf-pkix-pubkey-caps-02.txt>

"Internationalized Email Addresses in X.509 certificates", Alexey Melnikov, 7-Mar-11, <draft-ietf-pkix-eai-addresses-00.txt>

"DNS Certification Authority Authorization (CAA) Resource Record", Phillip Hallam-Baker, Rob Stradling, Ben Laurie, 7-Jul-11, <draft-ietf-pkix-caa-01.txt>

"KX509 Kerberized Certificate Issuance Protocol", Henry Hotz, 10-Jun-11, <draft-hotz-kx509-03.txt,.pdf>

XIV. Index

Résumé :

Le contenu de cette thèse porte plus spécifiquement sur les possibilités liées aux modifications de certains protocoles de l'Internet, en particulier le protocole SMTP, la mise en œuvre de spécifications peu utilisées, et les outils et méthodes envisageables pour garantir l'identification des parties de façon simple et transparente pour les utilisateurs.

L'objectif est de définir, d'une part une méthodologie d'utilisation de la messagerie pouvant assurer fiabilité et confiance, et d'autre part de rédiger les bases logiques de programmes clients et serveurs pour la mise en application de cette méthodologie.

Descripteurs :

«spam» ; pourriel ; e-mail ; confiance ; identification ; authentification ; SMTP ;

Title and Abstract :

The content of this thesis will focus primarily on opportunities related to changes in some Internet protocols, in particular SMTP, implementation specifications rarely used, and the tools and possible methods to ensure the identification of parties in a simple and transparent way for users.

The objective is to define, firstly a methodology for using the mail with reliability and confidence, and secondly to draw the logical foundations of client and server programs for the implementation of this methodology.

Keywords :

«spam» ; Junk emails ; confidence ; authentication ; identification ; SMTP