修士論文

メールヘッダへのID挿入による電子メールの高機能化に関する研究

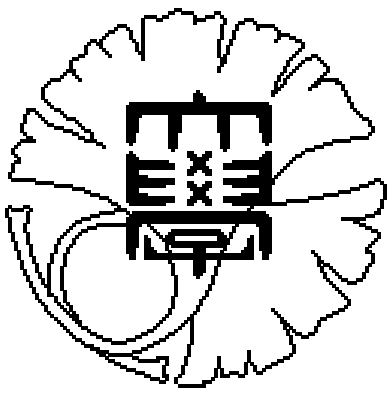
Functional Enhancement of E-mail
by Embedding ID in Mail Headers

2005 年 1 月 31 日

指導教官　相田 仁 教授

東京大学大学院 新領域創成科学研究科
基盤情報学専攻　36328

光田 智史



内容梗概

近年 SPAM メールが増え続けている理由の 1 つとして，現在の電子メールの配送プロトコルである SMTP に，受信したメールの送信元を認証する機能がないことがあげられる．電子署名で，End-to-End で送信元認証を行う技術は存在するが，これは各ユーザにソフトを導入する必要があるなど様々な問題があり，広く普及していない．

そこで本論文では，サーバ上，すなわちドメインレベルで送信元認証を行う手法を提案する．送信側のサーバは，ユーザから受け取ったメールに対して ID を生成し，それをデータベースに保存すると共にヘッダ内に埋め込んで送信する．受信側のサーバではそのヘッダが付いていた時には，送信元アドレスのドメインに対し，その ID の付いたメールを送信したことを問い合わせてから受信する．

またこの生成した ID を利用し，アンケートをメールで取った際に，返信メールを自動集計する機能についても提案する．アンケートを送信する人は，メールに拡張ヘッダをつけるだけで，回答する人は，そのメールに “返信ボタンを押す” だけで，この機能が利用できるように設計した．

目次

図目次

表目次

第1章

序論

1.1　はじめに

電子メールは，インターネットにおいて World Wide Web(以下，Web) と共に最も普及したサービスの一つであり，現在多くの人が使っている．[1]．ビジネスマンがパソコンで受信する電子メールは，2003 年の時点で一日平均 65.8 通にも達しており，間違いなく電子メールが業務コミュニケーションの中心になってきている [4]．またプライベートにおいても，ほぼ全ての携帯ユーザはインターネット接続サービスによって電子メールを利用している．

しかしこのように電子メールが広く利用されるにつれ，SPAM メールが深刻な社会問題となってきている．SPAM メールとは Web や NetNews などを通じて手に入れた不特定多数のメールアドレスに向けて，宣伝広告などを大量配信されたメールのことである（Fig. 1.1）．世界規模で見ると 2003 年に，全メールのうち SPAM がしめる割合が 50 %を超えたと言われ，増え続ける SPAM メールのために以前より電子メールを使わなくなったり，電子メールを信用できなくなってきた人も出てきている [2, 3].

Fig. 1.1: Ever-increasing SPAM.

SPAM メールがこのように横行している原因は SMTP(Simple Mail Transfer Protocol) という電子メールの送信プロトコルにあると言われている．SMTP では，メールは送信側から受信側に向けて一方的に送信され，受信側はただそれを受け取るという非常に単純なプロトコルである．この SMTP を使うことによって，メールは送信側から受信側へと一方向にのみ流れていく．

本論文ではこれを “電子メールの単方向性” と呼ぶ．この単方向性はメールの受信側が，送信側を信頼するというモデルの下に成り立っている．従来このプロトコルが設計された時代は，メールは研究者達の信頼できる間柄で使われることを想定していたが，その後インターネットが商用化され大規模になり，“いい人” ばかりでなく，“悪い人” も使うようになった．電子メールの単方向性は，裏を返せば電子メールの受信者はメールの送信元を確かめる手段がないということになる．そのため，SPAM 業者に SPAM メールを送信元を偽装して送ることを可能にしてしまっている．これは受信者に被害があるだけでなく，送信元を偽装された人もそのクレームメールなどで

大きな被害を被ることになる．

そこで本論文では，送信側のサーバは，メール送信の際に送ったメールの情報を保持しておき，受信側のメールサーバは送信側のメールサーバに対し，メールを送ったことを問い合わせて送信元を認証してから受信する “ハンドシェーク方式” のメール送信システムを提案する．具体的にはサーバ上で各メール固有の ID を発生させ，データベースに保存し，さらにその ID をメールヘッダに埋め込むことによって実現する．

また，電子メールの単方向性のため，メールをコミュニケーションツールとして使いづらい場合がある．その 1 つが，多人数に対してアンケートを取ることである．メーリングリストなどの機能を用いることによって，多人数にアンケートメールを送信することは，比較的簡単にできる．しかしそれらの返信メールは 1 通 1 通別々に送られてくるため，1 通 1 通に対して目を通さなければならず，非常に手間がかかるという問題がある（Fig. 1.2）．これは複数のメールを束ねる機能が存在しないからである．そこで本論文では，ハンドシェーク方式で発生させたメール固有の ID を利用し，アンケートの返信メールをメールサーバ上で集計して，その結果を一通のメールとして返す手法を提案する．

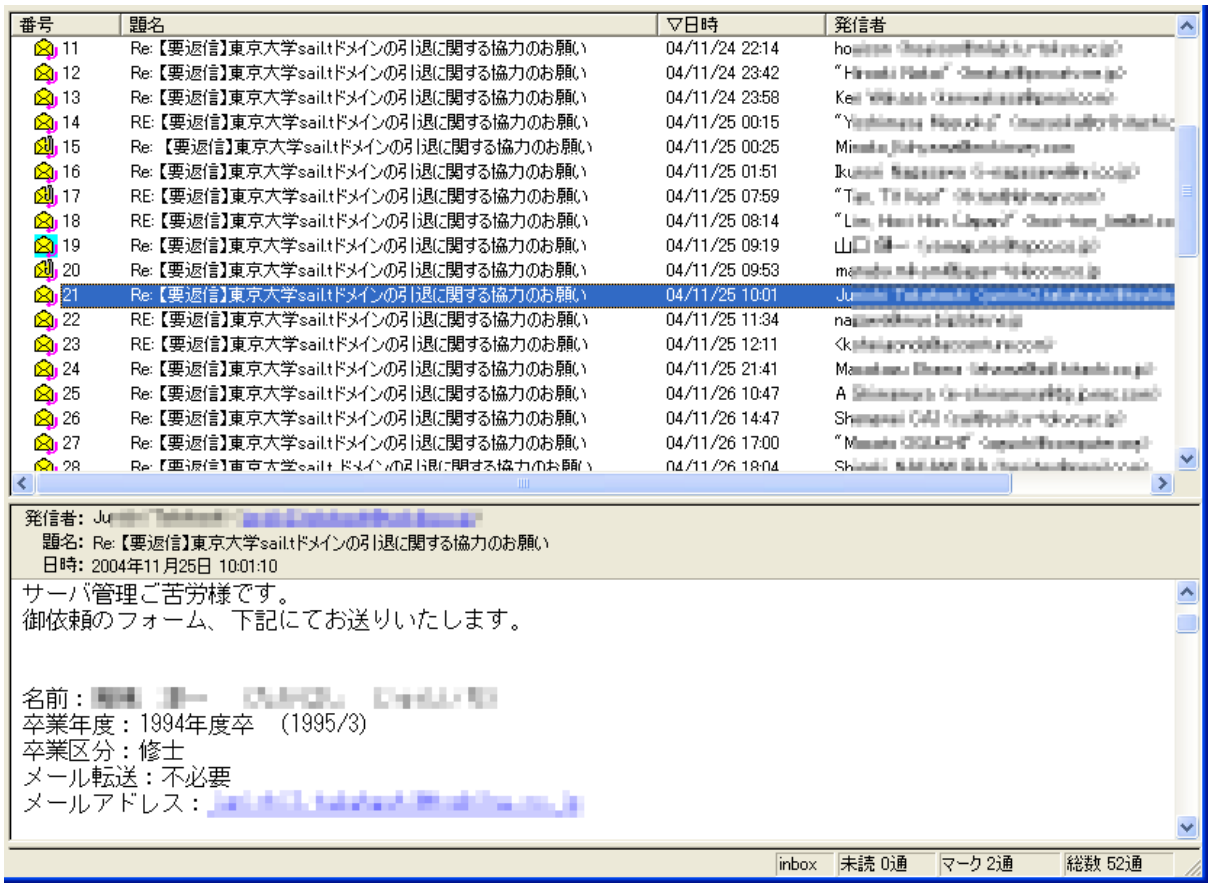


Fig. 1.2: Sea of reply mail in a questionnaire.

1.2　本論文の構成

本論文は，5 つの章から構成されている．以下に各章の構成を示す．

第 1 章

序論として，研究の動機，および本論文の構成について述べる．

第 2 章

SPAM メールの現状とその防止技術について説明する．

第 3 章

SPAM メールの送信元偽装問題に対処するため，ドメインレベルで送信元認証の一方式として，受信側のメールサーバが送信側のメールサーバに対し，メールを送ったことを確認してから受信するメール送信システムを提案し，その実装を行う．

第 4 章

複数のメールを一つに束ねる機能の一つとして，アンケートの返信メールを自動で集計する機能を提案し，その実装を行う．

第 5 章

本論文のまとめを行い，今後の課題について述べる．

第2章

研究の背景

2.1　はじめに

本章では研究の背景として，まず始めにインターネットにおける電子メールの送信プロトコルや配送の仕組みを説明する．そして近年増え続けている SPAM メールの現状を述べ，増加の原因を技術的観点から探る．さらに SPAM メールの踏み台にされないための技術，SPAM メールの受信を防止するための技術について紹介し，それらの検討を行う．

2.2　SMTP

2.2.1　プロトコル概要

現在のインターネットにおける電子メールでは SMTP(Simple Mail Transfer Protocol) というプロトコルが使われている．1982 年に RFC821 として公開されて以来様々な拡張がなされ，2001 年 4 月に RFC2821[5] が公開された．これは初期の SMTP(RFC821) や，その拡張である SMTP Extension(RFC1869) などを統合・整理した修正版となっている．

そのプロトコルは “Simple” と名がついているとおり，非常に簡単なものである．SMTP では一般的に TCP の 25 番ポートを使用し，1 行は CR+LF(0x0d+0x0a) で改行される．SMTP クライアントは 4 文字のコマンドを送り，SMTP サーバは 3 桁の数値とメッセージという形式で応答を返す．一般的なメッセージのコマンドシーケンスを Fig. 2.1 に示し，以下説明する．

まず最初に，クライアントがサーバに接続すると，サーバは応答コード 220 を用いて自分を名乗る．クライアントは HELO(EHLO) コマンドを用いて，自身を名乗り，次にサーバが応答コード 250 を返すことによって，セッションが開始される．ここまでをグリーティングという．これは互いに相手のサーバ名などを確認する作業である．

次にクライアントは MAIL FROM コマンドでメールの送信者を通知し，サーバは応答コード 250 を返す．そしてクライアントは RCPT TO コマンドでメールの宛先を通知する．サーバは，RCPT TO コマンドで伝えられたアドレスに送信できるかどうかを判断し，送信できる場合は応答コード 250 を返す．複数のユーザにメールを送る場合には，この RCPT TO コマンドは複数回繰り返されることになる．

これが終わるといよいよデータの送信である．DATA コマンドでメッセージの送信開始を通知し，サーバから応答コード 354 が返されたら送信データを送信する．送信データの終わりはピリオドだけの行を送ることで伝える．サーバから応答コード 250 が返されると，めでたく送信が完了するので，QUIT コマンドを送りセッションを終了する．QUIT コマンドにはサーバから応答コード 221 が返される．

尚，それぞれのコマンドに問題がある場合には，それに応じたエラーコードが返される．

2.2.2　7 ビットコードと MIME

原則的に SMTP では 7 ビットコードのテキストしか使用されない．これはもともと電子メールが英語圏で生まれたもので，ASCII コードが 0x00 から 0x7f までの範囲であることに起因する．もし 8 ビットコードのメールをそのままテキストとして送信してしまうと，SMTP サーバによっては 7 ビット分しか通さず，その結果 1 ビット分失われ文字化けを起こす可能性がある．そのため，日本語のメッセージを送信する場合は，EUC-JP や Shift_JIS は使えず，ISO-2022-JP というコードが使用される．ISO-2022-JP は 7 ビットで表現されており，ASCII 文字と漢字などの文字の切り替えにはエスケープシーケンスを用いている．

Fig. 2.1: SMTP sample session.

また，8 ビットコードや，音声，静止画，動画などのバイナリデータを 7 ビットに変換するエンコーディングの方法や，それらをアプリケーションや言語と関連付けてメッセージの中に構造化して格納する方法として，MIME(Multipurpose Internet Mail Extensions) という規格がある．MIME は RFC2045[6]〜RFC2049 などで公開されている．MIME を使用することにより，あらゆる種類のコンテンツをテキスト化して送信することができる．

Fig 2.2 は，本文にテキストが記載され，なおかつ “Trip.gif” という GIF ファイルを添付したメールの例である．“MIME-Version:” フィールドで MIME のバージョンを示し，“Content-Type:” フィールドでボディ部分の属性を，“Content-Transfer-Encoding:” フィールドでデータのテキストへの変換方法を示している．このように “Content-Type:” フィールドに multipart というメディアタイプを使うことで複数のアプリケーションデータをメールに添付して送信することが可能である．

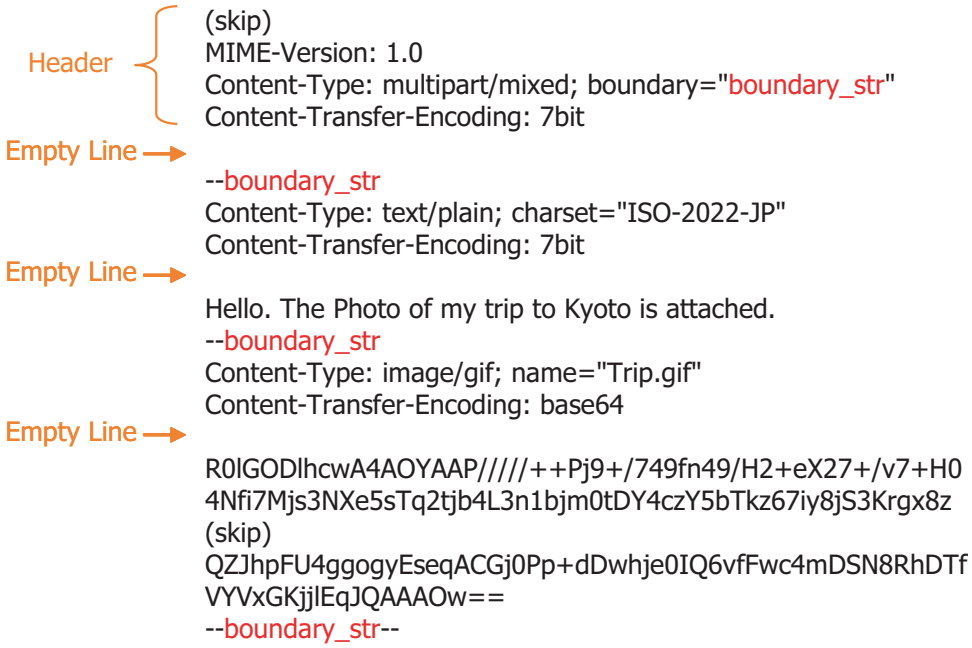


Fig. 2.2: Multipurpose internet mail extensions.

2.2.3　電子メールの配送

電子メールの配送の様子を Fig.2.3 に示す．電子メールの読み書きやメールサーバへの送信，サーバの自分のメールボックスからメールの受信などを行うソフトを MUA(Mail User Agent) と言う．Outlook Express や Winbiff などがこれにあたる．一方，MUA から送られてきたメールを相手先のメールボックスに届けるプログラムを MTA(Mail Transfer Agent) と言う．Sendmail や qmail などがこれに該当する．

MUA から送信されたメールはメールサーバの MTA へ SMTP を用いて送られる．それを受け取った MTA は，RCPT TO コマンドで受け取った宛て先メールアドレスのドメイン名から，DNS の MX レコードを使用して送信先 MTA を探し出し，SMTP で送信する[1]．宛て先 MTA は，ユーザのメールボックスにメールを配信する．すなわち，MTA はメール転送と，メール配信の二つの役割を果たしている．ユーザがメールサーバに届いたメールを手元の PC で見る際には，POP や IMAP といったプロトコルを使う．

ここで重要なことは，配送は SMTP コマンドの後のエンベロープアドレスに基づいて行われるということである．メールヘッダはメッセージデータの一部にすぎず，配送制御には使われない．そして，通常我々が目にするメールの送信元アドレスというのは，メールヘッダに書かれた “From:” フィールドを MUA が解釈して表示しているだけである．

[1] 送信先ホストが自分自身であった場合は，そのままメールボックスへ配信される

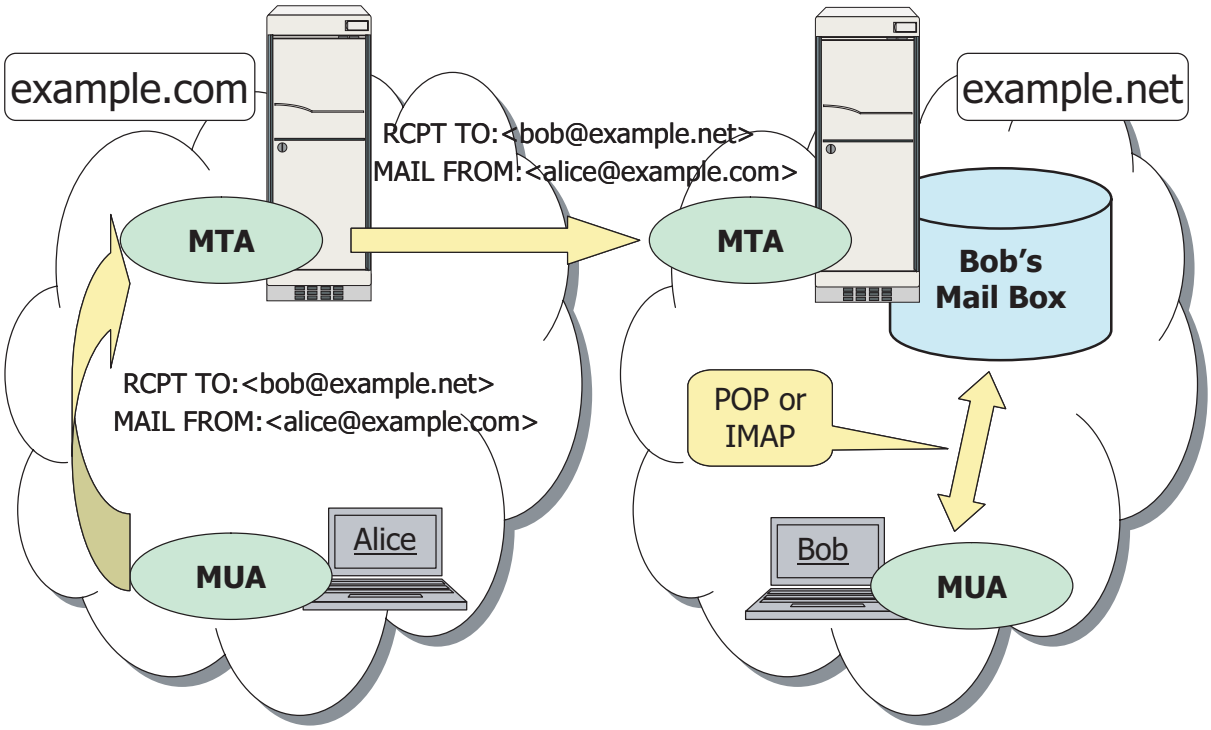


Fig. 2.3: Delivery of email messages.

2.3　SPAM メール

2.3.1　SPAM メールの現状

SPAM メールは電子メールの普及と共に増え続け，世界規模で見ると 2003 年についに全メールのうち SPAM メールが占める割合が 50%を超えたと言われている [2]．今現在はインターネット先進国であるアメリカなどの方が，より深刻な問題になっているが，日本でも携帯電話の SPAM メールがユーザを困らせ，様々な社会問題を引き起こしている．SPAM メールは，送信者側から見ると，広告の手段として他のタイプのダイレクト・マーケティングよりも短時間かつ低コストで送信できるため，返答率は低いとしてもはるかに効果のあるものである．一方受信者にとってみると，受信者の都合を考慮せず一方的に送られてくるこうしたメールは，余計な通信費やリソースを浪費する．SPAM メールを送るという行為は，一方的に電話でいうところのコレクトコールをかけること，従来の郵便システムでいうところの料金不足の郵便を送りつけることに相当する．よってその対応に貴重な時間とコストを費やさざるを得ない．さらに，アダルトメールなどによって精神的な苦痛も強いられる．これ以上 SPAM メールが増え続けると，大量の SPAM メールの中に大切なメールが埋もれてしまい，見落としてしまう可能性が一層高くなってしまう．

SPAM メールの影響はユーザの電子メールの使い方に，大きな影響を与えている．Pew Internet & American Life Project の 2003 年の調査報告 [3] によると，電子メール利用者の 25%は SPAM メールの増加のために電子メールの使用を減らし，52%は SPAM メールによって電子メールが信用できなくなったと言っている．また最近 SPAM メールが増加するにつれ，様々な SPAM フィルタが世の中に出回っているが，SPAM フィルタによって，いつの間にか SPAM メールだけでなく必要なメールまでもブロックしてしまっていたり，逆に他人に送ったはずのメールが SPAM

メールだと判断され削除されてしまうといった心配の声も上がっている．そして増え続ける SPAM メールによって電子メールが信用できなくなった人もかなりの割合に達してきている．

最近では SPAM メールの一種として，金融機関などの実在する企業に似せた Web サイトを作り，そこにリンクを張った電子メールを正規のメールを装って送信し，ユーザを誘導してクレジットカードの番号などの個人情報を盗み取るといったフィッシング詐欺の被害が非常に増えている [7]．

米ガートナー社の報告書 [8] によると，フィッシング詐欺に引っかかった人々は米国で約 180 万人にのぼるという．フィッシング詐欺の電子メールを受け取った経験がある人とは 5700 万人を超え，そのうち約 19 %，1100 万人近くが，偽サイトにアクセスしてしまった．さらに約 3 %，178 万人は，カード番号などを入力してしまっている．クレジットカード詐欺の被害額はこれまでに年間 12 億ドル相当にものぼっている．

SPAM メールはこのように多くの問題を抱え，それを防止するために数々の研究や議論がなされている [9–11]．ただ，SPAM メールと一言に言っても，その種類も送られる手段も様々であるため，それを防止する決定的な手法はなく，様々な分野からのアプローチが必要である．

まず政治面において，SPAM メールを取り締まる法律の制定が各国でなされている．アメリカでは 2004 年 1 月に CAN-SPAM(Controlling the Assault of Non-Solicited Pornography and Marketing) 法が施行された [12]．この法律においては，広告メールに返信用のアドレスや，メール配信を中止する Web ページへのリンクをつけることを義務付け，メールヘッダの偽造などを禁止している．日本においても 2002 年に「特定商取引に関する法律の改正」及び「特定電子メールの送信の適正化等に関する法律」などが制定されている [13]．しかし，いずれもあまり効果があがっていない．SPAM メールは一国内に留まらず国境を越えて送られ続けるため，政府機関や企業の国際的な取り組みが不可欠である．近年 OECD，IETF などの機関が，国際的な SPAM 対策に乗り出している [14, 15] が，未だに大きな効果が出ているとは言えない．

2.3.2　SPAM メール増加の原因

SPAM メールが増加している原因の 1 つとして，第 2.2 節で述べた SMTP プロトコルの欠陥が挙げられる．SMTP は，メールの受信側が送信側を信頼することが前提になっており，送られてきたメールの送信元を確かめる手段がない．すなわち，MAIL FROM コマンドの後に続く Return-Path[2]を偽ってメールを送っても，また Return-Path とメールヘッダの “From:” フィールドに何の関連性がなくても問題なく届いてしまう．つまり，メールというシステムは送信側から受信側に向けて一方的に送信され，受信側はただそれを受け取るしかないという “プッシュ型” のシステムなのである．そのため，メールを受信したユーザが MUA で “From:” フィールドを見て送信元を調べても，それが本当にそのメールアドレスの人から来たのか確かめられないため，SPAM 業者は SPAM メールやフィッシング詐欺メールを，送信元を偽装することによって身元を隠して送ることが可能となってしまっている．

従来このプロトコルが設計された時代は，ネットワーク上のコンピュータはせいぜい数百から数千のオーダーであり，メールを使用するのは研究者たち，すなわち信頼できる間柄を想定していたため，問題ではなかった．しかし，今現在の大規模かつ商用化されたインターネットにおいては，この SMTP が SPAM メールの観点から見て，大きな “セキュリティホール” となってしまっている．

[2]MAIL FROM コマンドのアドレスはエラーメールなどを返すときなどに使われるためにこう呼ばれる．

2.4　第三者中継防止技術

SPAM メールの多くは，不正な第三者中継と呼ばれる方法で送信されている．第三者中継とは，実際の送信者が無関係の第三者のメールサーバを不正中継してメールを送信することである．第三者メールサーバを中継することにより，送信者は匿名または身元を偽ってメールを送信できる．そのため，中継に使われたメールサーバが，スパム送信者へのクレームが誤って寄せられる等の被害を被るだけでなく，偽装された発信者アドレスの人などにも迷惑がかかることになる．そのため，最低限メールサーバには，第三者中継を許可しないという設定が必要である．

そのために一番確実な方法は，社内専用サーバなどにおいて，LAN 内にあるコンピュータからのみメールの送信を許すという設定である．しかしこの場合，出張中のサラリーマンが出張先から本社のメールサーバを用いてメールを送信するなどということができなくなる．この問題を解決するため，どこでもメールが送信でき，かつ第三者が勝手にメールサーバを使用して SPAM メールを送信しないようにするための技術について紹介する．

2.4.1　SMTP-AUTH

SMTP-AUTH は RFC2554 で定義されている [16]．SMTP にユーザの認証がなかった問題を解決するため，AUTH コマンドを用い，SASL（Simple AUthentication and Security Layer）と呼ばれる汎用的な認証メカニズムによってログインを可能にしている．SMTP-AUTH においては，PLAIN，LOGIN，CRAM-MD5，DIGEST-MD5 などのメカニズムが一般的である．

PLAIN（RFC2595），LOGIN　通常の平文による認証方式である．SSL などによる暗号化の利用が前提になる．

CRAM-MD5（RFC2195）　CRAM とは Challenge-Response Authentication Mechanism のことである．まず “Challenge” と呼ばれる乱数文字列がサーバからクライアントに送られる．クライアントはこれとパスワードから，MD5 によってメッセージダイジェストを計算し，サーバに送り返す．サーバはクライアントのパスワードを記憶しているため，同様の暗号化をして，その結果とクライアントから返された暗号とを比較することでユーザの認証を行なうことができる．パスワード自身がネットワーク上に流れないため，安全性が高い．

DIGEST-MD5（RFC2831）　CRAM-MD5 の欠点である辞書攻撃や総当り攻撃に対する対処とともに，Realm（ログイン領域）や URL の指定をサポートしている．

2.4.2　POP before SMTP

ISP の運営する MTA でよく使われている方法である．POP before SMTP においては，SMTP のユーザ認証に POP の認証を用いる．ユーザがメールを送信したい場合は，まず POP でログインを行う．SMTP サーバは，POP サーバに認証の通ったユーザの IP アドレスのみ一定期間の間，接続を許可する．

これにより第三者が勝手にメールを送信することを防ぐことができるが，SMTP サーバと POP サーバの機能が連動しなくてはならないことや，IP アドレスの詐称が技術的に可能であることなどから SMTP-AUTH が普及するまでのつなぎの技術であると考えられている．

2.5　SPAM メール受信防止技術

SPAM メールの受信を防止するための手法としては，以下のようなものが挙げられる．

2.5.1　RBL

RBL(Realtime Blackhole List) という第三者中継ホストを登録したデータベースを参照して SPAM の可能性が高いメールを排除する手法である．MAPS[17] や ORDB[18] などが有名である．このデータベースを使うと，第三者中継の SMTP サーバから送信されたメールをすべて拒否するような設定が可能となり, SPAM メールの受信量を減らすことが可能となる．ただし勝手に自分のサーバが登録されると，メールが送信できなくなってしまうことや，データベースのチェックに時間がかかることなど，課題も少なくない．

2.5.2　コンテンツフィルタリング

MUA の SPAM フィルタ機能などにより，SPAM メールによく見られるフレーズを検索し，削除する．単純に指定した文字列で判定するものから，ベイジアンフィルタのような，過去の SPAM メールとそうでないメールの蓄積から，新たなメールが SPAM メールかどうか判定するようなものまで，様々なアルゴリズムが盛んに研究されている [34–36]．大まかな分類を Table 2.1 示す．

Table2.1: SPAM filter classification.

分類	説明	特徴
指定文字列によるフィルタ	繰り返し届くことの多い SPAM メールの文面を登録しておく．	手軽であるが，特定の SPAM メールにしか効果がない．
ヒューリスティックルール	SPAM メールの送信元や，文面などの特徴をルールとして膨大なデータベースに集め，利用する．	指定文字列に比べればある程度効果は大きいが，SPAM メールの特徴の変化に対応していけない．
ベイジアンフィルタ	過去に起きた事象の確率を利用して未来を予測する学習型フィルタである．	始めに学習が必要．SPAM メールによっては，ニュース記事やまったく無意味な文字列を本文の最後に付けることにより，単語の出現頻度をかく乱させるものがあるので，精度が落ちる．最近の出会い系サイトへ誘導する SPAM メールは，ごく普通の文章でかつ短いため，SPAM メールとして判定させるのは非常に難しい．

実際にはこれらを組み合わせて使うことで，ある一定の効果はあるが，あるアルゴリズムが普及すると SPAM 業者はそれに対抗した手法を取るといったように，研究者と SPAM 業者のいたち

ごっこのような状況が続いている．そして最近の SPAM メールは，知人を装ったり仕事の連絡を装ったりするなど手口が巧妙であるため，フィルタリングの精度はなかなか高くならない．そして SPAM フィルタリングの一番の問題点は，フィルタリングの精度を高くしようとすると SPAM でないメールが SPAM メールであると判断されてしまう危険性が高くなってしまうということである．

2.5.3　電子署名

SPAM メールの多くは送信元アドレスを偽装している [19]．この偽装を検出することができれば，正規のメールか SPAM メールかの見分けがつく．送信元アドレスを認証する究極の方法は，PGP，S/MIME などのように公開鍵暗号方式を用いてメールに電子署名をつけることである [20]．これらは，メッセージのハッシュを秘密鍵で暗号化してから送信する．メールの受信者は送信者の公開鍵で復号することでそのメッセージを作成した人を確認することが出来き，End-to-End でメッセージの完全性が保証される．

PGP と S/MIME の大きな違いは公開鍵の配布の方法である．公開鍵暗号方式では，受け取った相手の公開鍵が本当に正しいものであるかを確認する仕組みが必要である．

PGP では,「友達の友達は友達」というようにユーザがお互いに信頼関係を結び，その輪を広げていく．鍵の有効性については，各ユーザが判断する．具体的には，自分が良く知らない B さんの公開鍵を受け取ったときに，そこによく知っている A さんの署名がついていれば「A さんは信用できるから，これは間違いなく B さんの公開鍵である」と判断できる．すなわち信頼の輪を気づいていく方式である．そのため PGP は，一般的な電子メールの使われ方として考えられる不特定多数の人とやりとりするような場合には不向きである．ただ，PGP は認証局という第三者的存在を必要としないので，導入が容易であり，友達同士などの閉じたコミュニティの中で使うときに威力を発揮する．そのため，電子署名をしてメールの送信元を認証するといった目的よりも，部外者に情報をもらさないようにするために暗号化して送るといった目的によく使われる．

一方，S/MIME では，鍵の配布に PKI（Public Key Infrastructure）を使っている．これは，Fig. 2.4 のように，認証局という第三者機関が，公開鍵とその持ち主を保証する方式である．認証局は通常階層的になっており，ユーザはルート認証局を信頼することが前提になっている．ユーザ（例えば A さんとする）は下位のある認証局に公開鍵を提出し，その認証局は A さんの公開鍵に署名を付けた公開鍵証明書を発行する．ルート認証局は下位の認証局の公開鍵に署名をつけた公開鍵証明書を発行する．A さんの公開鍵証明書を受け取った人は，その証明書を発行した認証局が信頼できる認証局であることをルート認証局から辿ることによって確認し，受け取った公開鍵が確かに A さんのものであることを確認することができる．

しかし S/MIME は，PKI(公開鍵基盤) を整備し，1 人 1 人のユーザに導入する必要があること，暗号を用いているので，とても複雑でエラーが起こりやすく，秘密鍵が盗まれた時にそれを検出するのに時間がかかることなどから近い将来に完全に実現される可能性は極めて少ないと考えられている．

2.5.4　ドメインレベルでの送信元認証

2003 年秋から，LMAP(Lightweight MTA Authentication Protocol)[21–27] という送信元アドレスをドメインレベルで認証するプロトコル群が提案され，2004 年 3 月に，IETF は MARID ワーキンググループ [15] を設立し，プロトコルの標準化に向けて盛んに議論している．現状の SMTP

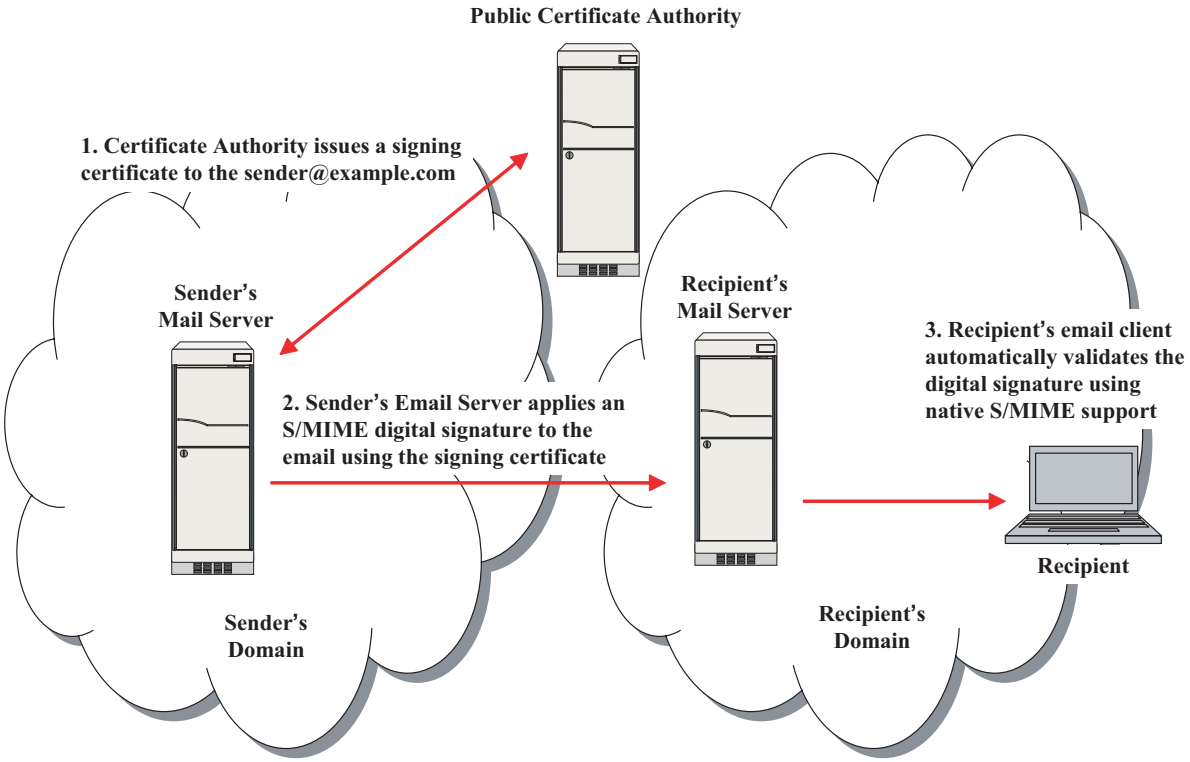


Fig. 2.4: S/MIME.

では，メールを “受信する”MTA の IP アドレスだけを DNS の MX レコードに書いておくが，これらはさらにメールを “送信する”MTA の情報も DNS に登録しておくことで，メールを受信したサーバが，送信元ドメインの権限のある IP アドレスからメールが送られてきているかを調べることを可能にしている．これらの目的とするところは，送信元 MTA(SMTP クライアント) を IP アドレスで認証することであるため，IP アドレスの偽装や DNS Spoofing までは対応できない．

以下，2004 年に入って主に研究されてきた技術について説明する．

SPF

SPF(Sender Policy Framework)[28] は America Online(AOL) が支持する技術で，その概要を Fig.2.5 に示す．受信側の MTA は送信元のエンベロープアドレスである Return-Path のドメイン名と SMTP クライアントの IP アドレスの対応を取ることにより認証を行う．例えば DNS に

```
example.com. TXT "v=spf1 mx -all"
```

と書いた場合，SMTP クライアントの IP アドレスが，example.com の MX レコードのホストの IP アドレスリストの中に存在したら SPF 認証を通過することになる．SPF ではエンベロープアドレスに基づいて送信元 IP を認証しようとしているが，通常 “.forward” などによってメールを転送した場合，Fig. 2.6 のように Return-Path は変えられずにに送られる．そのため，転送する MTA の IP アドレスと Return-Path の不整合が生じてしまう．

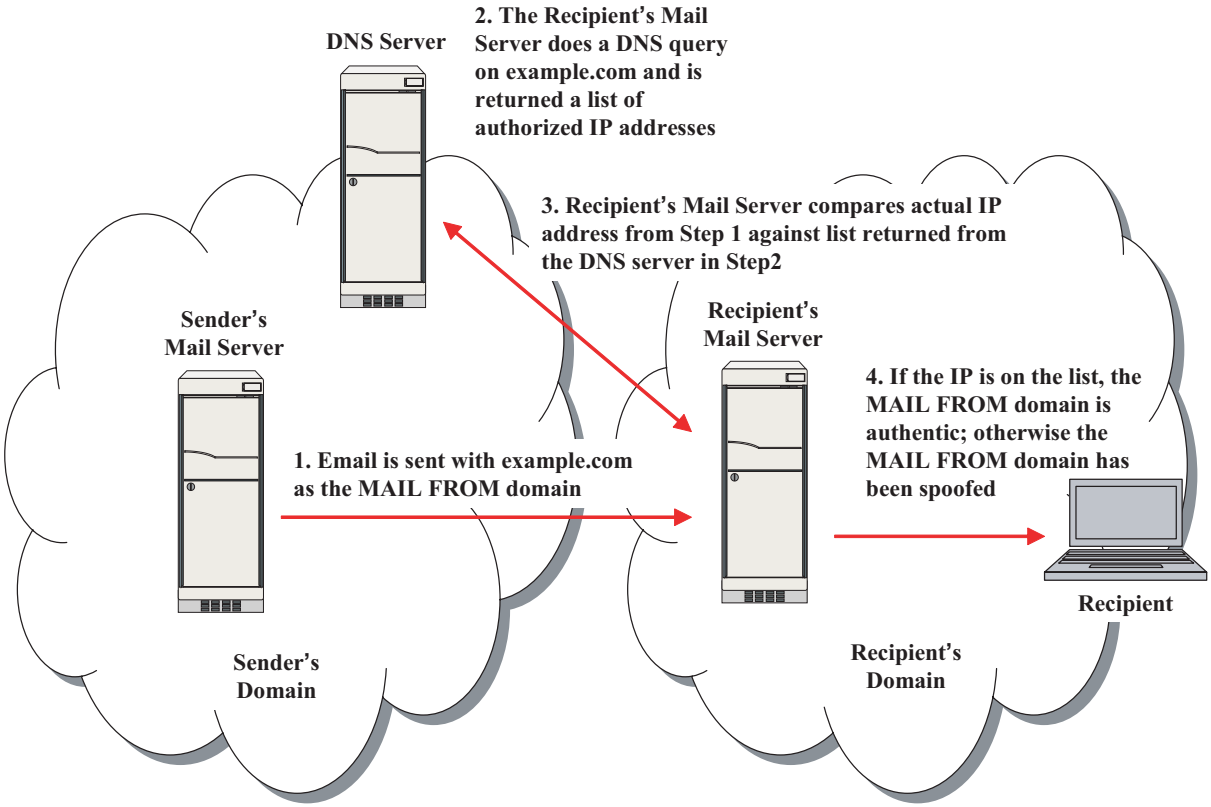


Fig. 2.5: Sender Policy Framework.

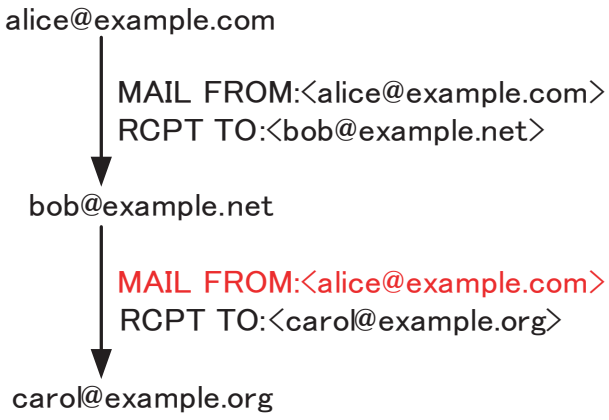


Fig. 2.6: Problem of mail forwarding.

Caller ID for E-mail

Caller ID for E-mail は Microsoft が提唱している技術で SPF と異なり，ヘッダの送信元アドレスを認証する．Caller ID for では，DNS に XML 形式で情報を発行するため，より複雑な処理が可能となっている．その概要を Fig.2.7 に示す．

Caller ID for E-mail では送信元アドレスとは，RFC2822[30] に定められた仕様に従い，“Resent-Sender:” フィールド，“Resent-From:” フィールド，“Sender:” フィールド，“From:” フィールドの順に優先すると定義している．しかし，通常 MUA で表示されるのは “From:” フィールドだけであり，“From:” フィールドに実在する企業を騙ったフィッシングメールなどは防止しづらいと考えられる．

またメールを転送する場合には，転送メールの送信元と IP アドレスの不整合が起こらないように，送信元アドレスを書き換えなければならないという問題がある．

SPF と Caller-ID for E-mail は 2004 年 6 月に統合されて，Sender ID[31] という 1 つの仕様として IETF に提出された．しかし Microsoft 社が自社部分の技術に厳しいライセンス条件を課しており，標準として普及するかどうかは未定である．

DomainKeys

SPF と Caller ID for E-mail は SMTP クライアントの IP アドレスを認証していたが，別の送信元認証として，DomainKeys という電子署名を使った技術も Yahoo!により提案されている [32]．その概要を Fig.2.8 に示す．DomainKeys では，送信側は DNS に公開鍵を保存しておき，MTA がメールのメッセージにハッシュをかけたものを秘密鍵で署名し，それをヘッダを埋め込んで送信する．受信側では公開鍵を取ってきて署名が本物であるかどうかを確認することができる．

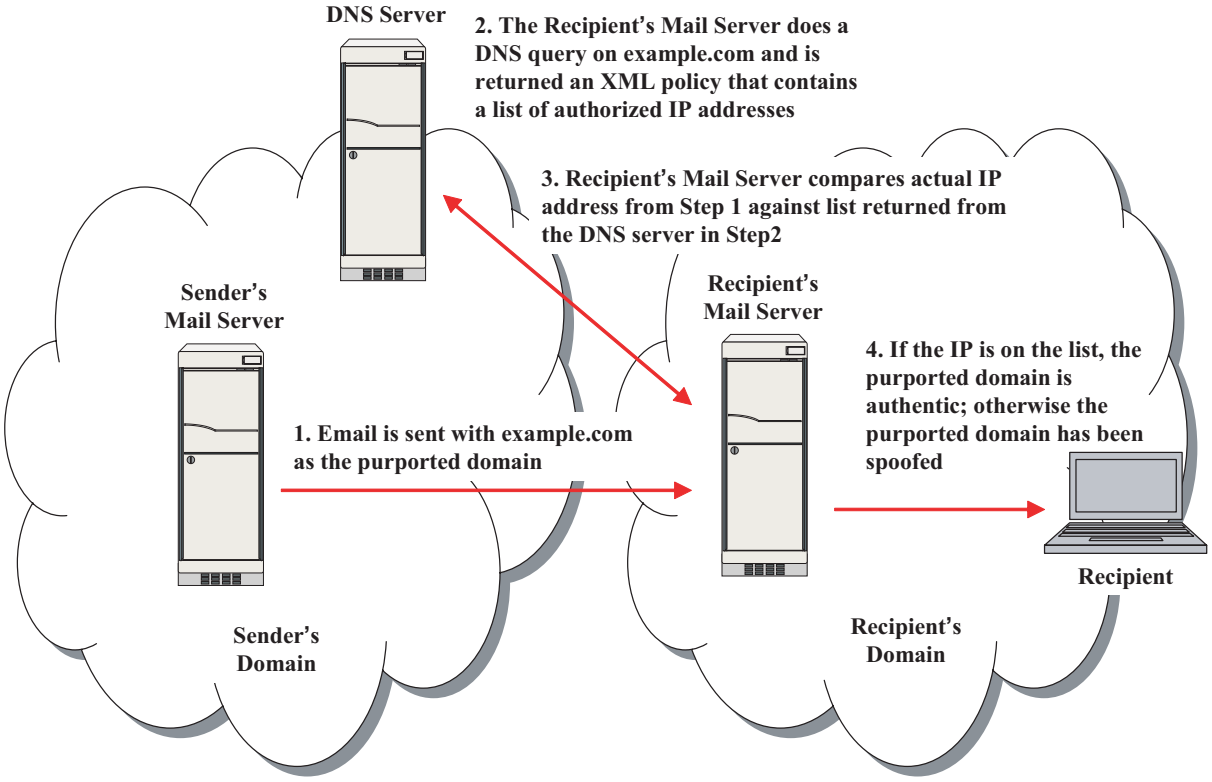


Fig. 2.7: Caller ID for E-mail.

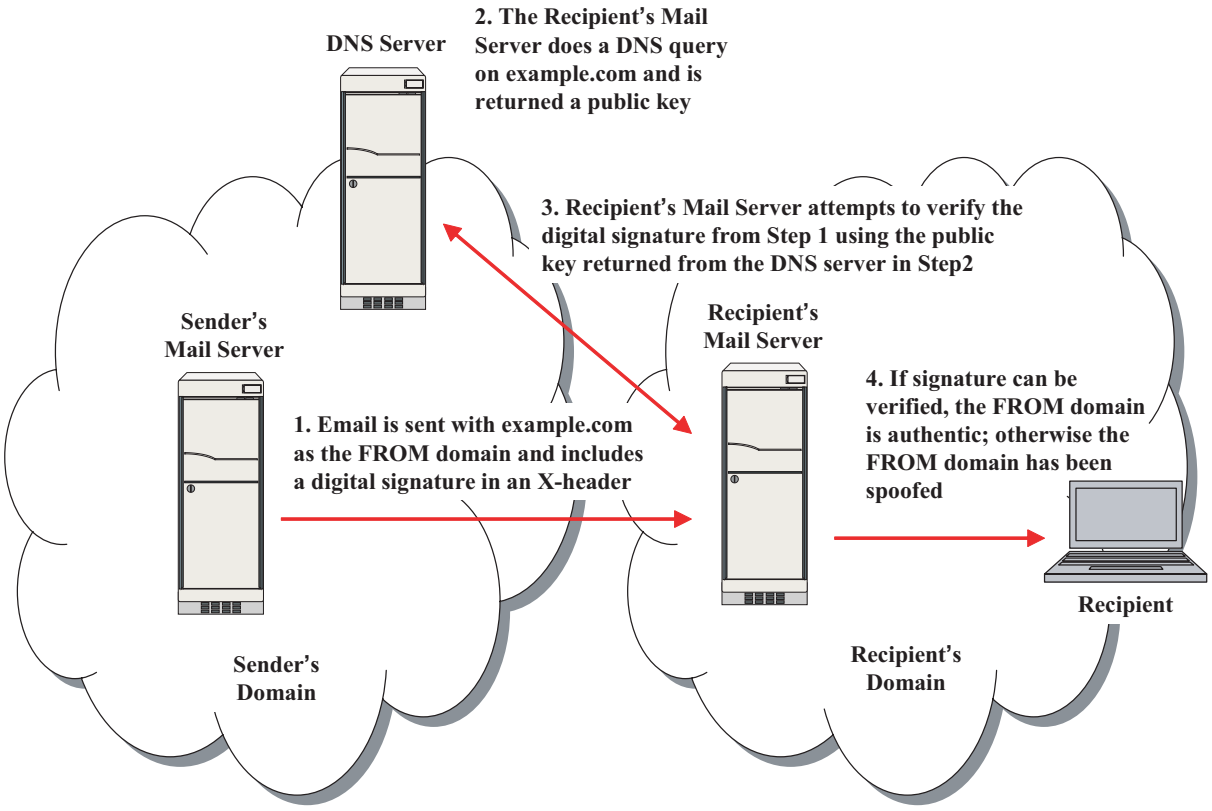


Fig. 2.8: DomainKeys.

第3章

送信元認証における
ハンドシェーク活用の提案

3.1　はじめに

第 2.5.4 節でドメインレベルでの送信元認証について述べてきたが，現在主流の SPF や Sender ID は，ラストワンホップの SMTP クライアントの IP アドレスを認証している．そのため，転送されて送られてくるメールや，ML などから配られるメールの大元の送信元アドレスを認証することができないという問題がある．また，電子署名を使った手法が近い将来に広く実現されることも想定し難い．

そこで本論文では送信側のサーバが送ったメールの情報を保持しておき，受信側のメールサーバは送信側のメールサーバに対し，メールを送ったことを問い合わせて確認してから受信するメール送信システムを提案する．本システムでは，様々なサーバを経由してメールが送られてきても，送信元アドレスのドメインに対して問い合わせるため，中間に本システムに対応してないサーバが存在していても，送信元と最終的な宛先の両ドメインのエッジサーバが対応していれば認証できるという利点がある．

3.2　SMTP-AUTH の実装によるユーザ名の認証

本システムでは第 2.5.4 節でのドメインレベルでの送信元認証について，もう一歩踏み込み，送信側のサーバが送信元アドレスのユーザ名を認証する仕組みを導入する．そのため全ての “善良な” サーバは SMTP-AUTH を実装し，ユーザの認証を行っているとする．ただし現在の SMTP-AUTH は，一旦認証を通過すると，そのユーザ名に関係なくどのような送信元を名乗ってメールを送信できてしまう．本システムでは，SMTP-AUTH のユーザ名はメールアドレスのユーザ名でなければならないとし，そうでないメールはメールの送信を拒否する．

これにより，Fig. 3.1 のようにメールを受信したユーザは，送信元のドメイン名が信用できる時は，ユーザ名まで含めて送信元が確認できることになる．ただ当然のことながら，送信側のサーバ自体が乗っ取られてしまった場合などは，確認することはできない．

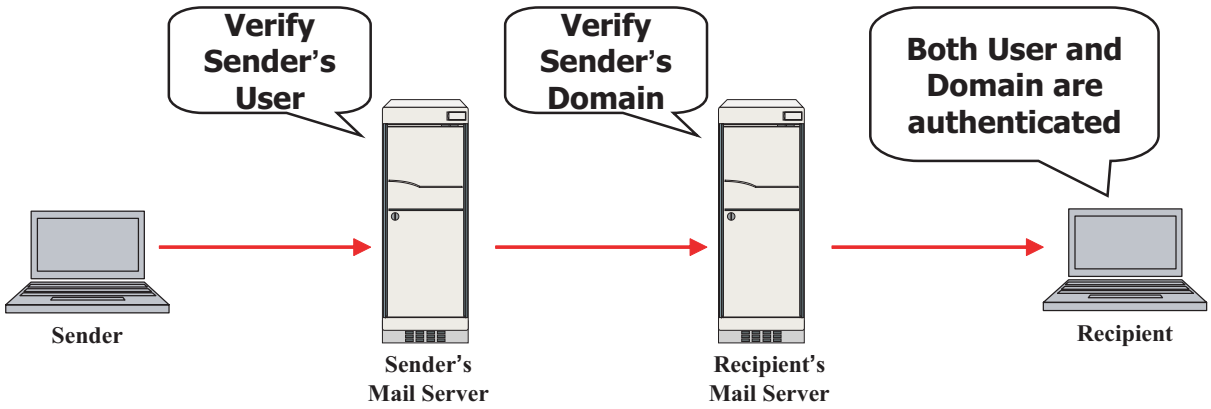


Fig. 3.1: Two-step authentication.

3.3　概要

概要を Fig. 3.2 に示す．送信側のメールサーバは，ユーザからメールを受け取ったら，各メールにユニークな ID を発生させ，それをデータベースに保持し，さらにヘッダ内に挿入して受信側のメールサーバへ送信する．受信側では送信元サーバをドメイン名を元に割り出し，そこに向けてその ID を付けたメールを本当に送信したか問い合わせる．本提案手法は，送信側と受信側の MTA が協調して動作を行う必要がある．どちらか片方が対応していない場合は従来通りのメールとして送受信される．

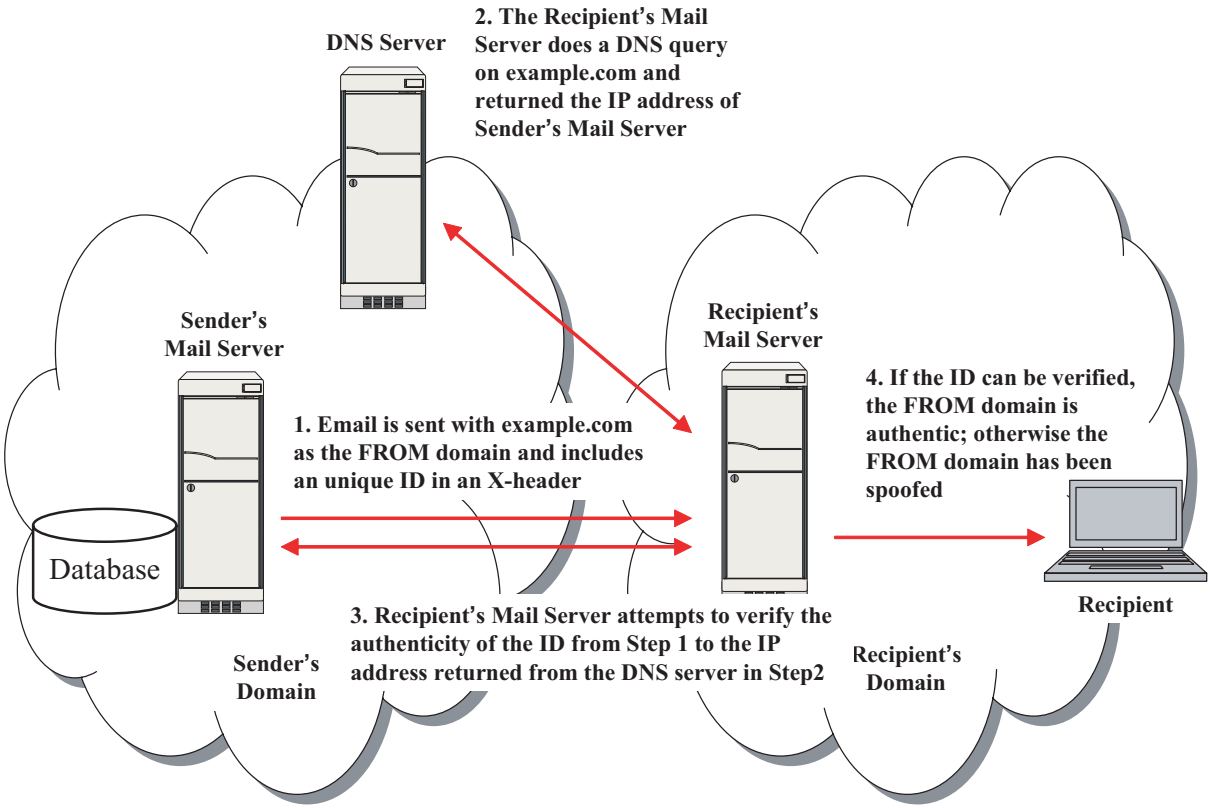


Fig. 3.2: Summary of Handshake method.

3.4　設計方針

前節の Fig. 3.2 を実現するための具体的な設計について記す．

3.4.1　拡張ヘッダの使用

Winbiff，Becky!，AL-Mail などの MUA では，メールに任意のヘッダをつけることができる (Fig. 3.3)．

“X-” で始まるヘッダはユーザで自由に定義してよい [30] ため，ユーザが送信元を信頼してほしいような責任のあるメールを送る場合は，

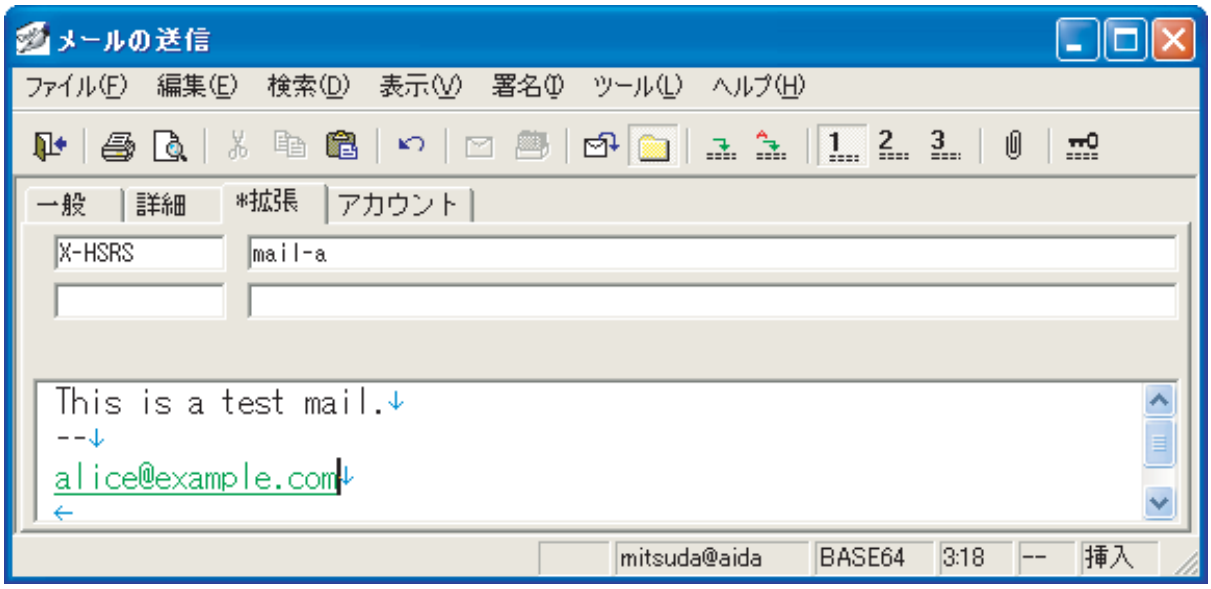


Fig. 3.3: Extention header(Winbiff).

```
X-HSRS: <サーバ名>
```

というヘッダをつけてメールを送ることとする．予めメールサーバの管理者は，そのドメインのユーザに問い合わせに応答するサーバ（以下，Reply Server）の名前を知らせておくこととする．Fig. 3.3 の例では，example.com ドメインの Reply Server は，mail-a である．

もしそのヘッダがついていた場合は，MTA は 1 つ 1 つのメールに固有な ID を作成し，

```
X-HSID: <ID>
```

というヘッダをつけて宛て先 MTA に転送する．

それを受信した側は送信元に問い合わせて確認が取れたら

```
X-HSResult: Success
```

というヘッダをつけてユーザのメールボックスに配信する．

3.4.2　送信元アドレスの定義

本論文においては，送信元アドレスとはエンベロープではなく，最終的にユーザの MUA に表示されるヘッダの “From:” フィールドと定義する．“From:” フィールドのドメイン名をメールの送信に責任のあるドメインとし，そのドメインの管理する DNS に Reply Server の IP アドレスを載せておく．

3.4.3　Reply Server 名の割り出し

受信側が ID を問い合わせるサーバ名は「“X-HSRS:” フィールドのホスト名」,「.(ピリオド)」,「“From:” フィールドのドメイン名」をつなげたものとする．例えばヘッダ情報が

```
X-HSRS: mail-a
From: alice@example.com
```

であった場合には，mail-a.example.com を DNS 検索する．

これにより，データベースを共有していれば，メールの送信とデータベースへの問い合わせの応答を別のサーバにすることができる．また example.com の送信サーバが東京 (192.168.1.1) と大阪 (192.168.1.2) に 2 台あり，別々のデータベースを使用している場合や，他のドメインである example.net の送信サーバ (172.20.1.1) を使わせてもらう場合には，DNS に以下のように記述しておけばよい．

```
reply-tokyo.example.com. A 192.168.1.1
reply-osaka.example.com. A 192.168.1.2
reply-net.example.com. A 172.20.1.1
```

メールの管理者は，東京のサーバを使うユーザには “reply-tokyo”，大阪のサーバを使うユーザには “reply-osaka” を “X-HSRS:” フィールドの中身としてを使ってもらうよう指示しておく必要がある．

3.4.4　メール固有の ID の求め方

送信側の MTA は，送るメール 1 つ 1 つに固有の ID をつけ，それを後からの問い合わせに答えるためにデータベースに保存する．ただし，その ID はただの乱数ではいけない．Fig. 3.4 のように，悪意のあるユーザが，送信されるメールの内容を途中で覗き見した場合に SPAM メールにそれと同じ ID を付けて，他の人に送信されてしまうからである．すなわち，第三者に真似られないような ID を生成する必要がある．

そのため，本論文では From，To，Subject，Date のフィールドをつなげたものを MD5 によりハッシュをかけたものを ID とする．

```
<ID> = MD5( From || To || Subject || Date)
```

そして受信側の MTA では，この ID を再計算することで各フィールドが偽られていないことを確認する．

一方向性のハッシュ関数を用いることにより，同じ ID を作るには，この 4 フィールドが全て同じメールを作成しなければならないため，受信したユーザは容易に Replay Attack に気づくことが可能である．

3.4.5　ID の問い合わせ方法

メールの受信側から Reply Server に向けての ID の問い合わせは，拡張した SMTP コマンドを使用し，SMTP セッションとして行う．具体的には，以下のコマンドを打つことによりその ID がリストに存在するか問い合わせる．

```
XRPY <ID>
```

Reply Server は XRPY コマンドを実装し，これが実行された場合はデータベースを検索し，ID がリストに存在すれば応答コード 250 を返す．存在しない場合は応答コード 554 を返す．

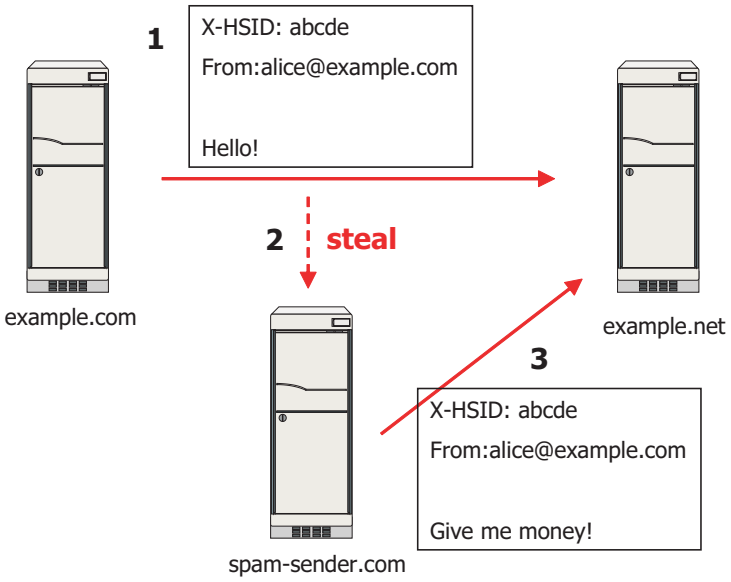


Fig. 3.4: Replay attack.

3.4.6　結果の表示

受信側サーバは問い合わせの結果，ID がリストに存在することが確認できたら

```
X-HSResult: Success
```

というヘッダを，仮に失敗した場合は

```
X-HSResult: Error
```

をつけて，ユーザのメールボックスに配信する．後者のメールは送信元が偽装された SPAM メールである可能性が高い．

3.4.7　ID リストの消去

送ったメールが複数に転送される可能性があるため，問い合わせがあった ID をすぐにデータベースのリストから消去することはできない．ID は，作成から 7 日間たったものはリストから消去するようにする．

3.5　全体の処理の流れ

alice@example.com が bob@example.net に向けて，自分のドメインの MTA(mail-a.example.com) から相手先 MTA(mail-b.example.net) を経由してメールを送信する場合を例に，Fig. 3.5 を用いて説明する．尚，Fig.3.5 においてはメールの送信サーバと問い合わせに答えるサーバは同一のものとしてある．

1　まず，alice の MUA は mail-a に接続をし，“X-HSRS: mail-a” というヘッダをつけたメールを bob 宛てに送信する．

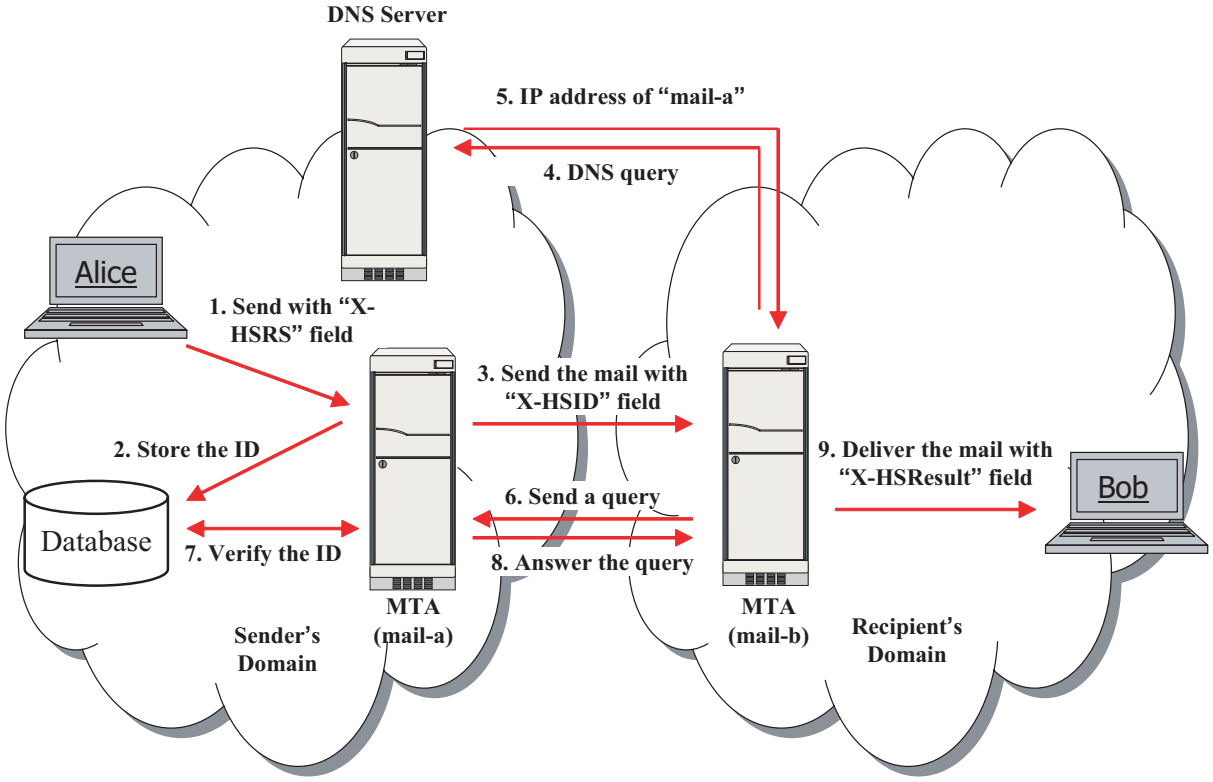


Fig. 3.5: Message flow in Handshake mail system.

2 mail-a は，受け取ったメールに “X-HSRS:” フィールドが存在した場合，ID を作成しデータベースに保存する．

3 その後 mail-a は，“X-HSID: <ID>” ヘッダを頭につけ，宛て先メールアドレスから mail-b を探し出し，SMTP を用いて mail-b にメールを送る．

4,5 mail-b は，受け取ったメールに “X-HSID:” フィールドが存在した場合，ID を問い合わせるサーバ名を割り出し (詳細は後述)，DNS を検索する．

6 mail-b は，5 で得た IP アドレスに接続し，ID がリストにあるかを尋ねる．

7,8 問い合わせのあったサーバは，データベースを検索し，結果を返す．

9 mail-b は，問い合わせの結果を “X-HSResult:” フィールドとして頭につけて，ユーザのメールボックスに配信する．

3.6　実装

3.6.1　qmail

UNIX 系の OS 上でよく使われている MTA のプログラムとしては，sendmail と qmail が挙げられる．sendmail はローカル配信やリモート配信など全ての処理を，“sendmail” という 1 つのプロセスで管理しているのに対し，qmail は，メール配送にかかわる処理を複数のプログラムに分割しており，各プログラムが単純で，改変しやすいという特徴がある．そこで本システムはこの qmail を用いて実装することにした．

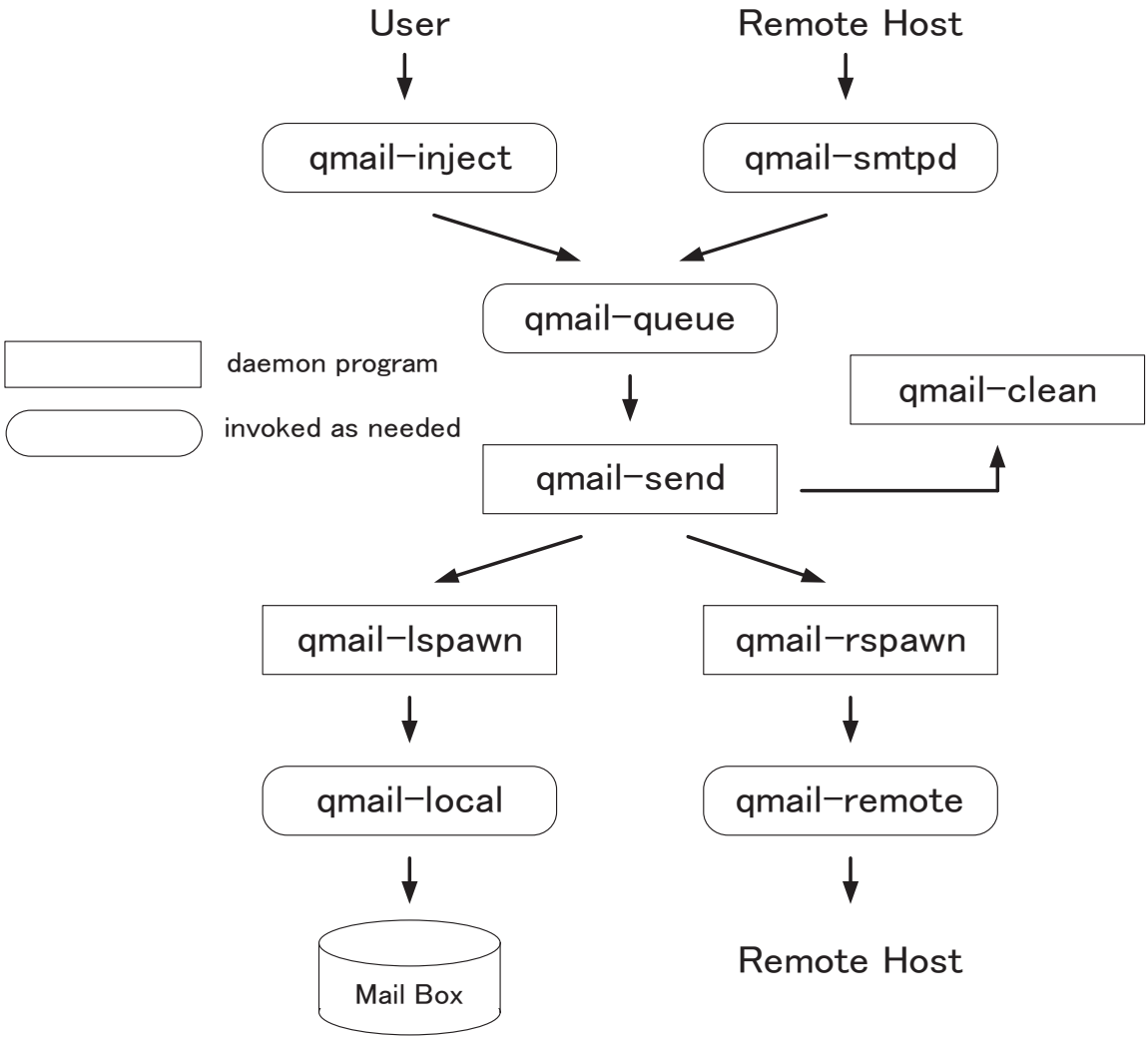


Fig. 3.6: Process flowchart of qmail.

qmail は，D. J. Bernstein 氏によって提供されており，2005 年 1 月現在の最新バージョンは 1.03 である [37]．qmail の内部構造を Fig.3.6 に，その各プログラムの役割を，Table 3.1 に示す．

3.6.2　PostgreSQL

本システムは，データベース管理システムとして，オープンソースソフトウェアの PostgreSQL を使用した．PostgreSQL は，標準的な問い合わせ言語の SQL をサポートしており，機能が豊富で信頼性が高く，パフォーマンスに優れている．UNIX プラットフォーム上で動作し，FreeBSD や Linux などの UNIX 互換のシステム上でも動作する．

C 言語から PostgreSQL にアクセスするために，libpq ライブラリを使用した．libpq ライブラリを使う PostgreSQL クライアントアプリケーションは，全て libpq の提供する関数を定義するヘッダファイルをインクルードし，-lpq を使ってそれらの関数のコードを含むライブラリをリンクする必要がある．

Table3.1: Internal structure of qmail-1.03.

qmail-inject	preprocess and send a mail message
qmail-smtpd	receive mail via SMTP
qmail-queue	queue a mail message for delivery
qmail-send	deliver mail messages from the queue
qmail-clean	clean up the queue directory
qmail-lspawn	schedule local deliveries
qmail-local	deliver or forward a mail message
qmail-rspawn	schedule remote deliveries
qmail-remote	send mail via SMTP

3.6.3 実装方法

1 つの MTA が，送信サーバと受信サーバの両方を役割を果たしている場合は,「受け取ったメールヘッダに，X-HSRS だけ存在したら X-HSID をつけて送信（転送），X-HSID も存在したら ID を送信側に問い合わせて受信」という処理が必要である．これらは，ヘッダの解析を行うことが目的なので，メールメッセージを配送キューに格納するプログラムである qmail-queue にラッパをかぶせることで実現した．処理の流れ図を Fig. 3.7 に示す．

SMTP-AUTH は，qmail-smtpd のソースである qmail-smtpd.c に，[38] から入手したパッチを宛てることで実装した．CRAM-MD5 にも対応させるため，cmd5checkpw[39] も用いた．SMTP-AUTH を通ったユーザ名は，“AUTHUSER” という環境変数に保存するようにした．この値をヘッダを解析する qmail-queue のラッパ関数から参照することで，“From:” フィールドのユーザ名と比較が可能となる．XRPY コマンドも，qmail-smtpd.c を改変することによって行った．

データベースは，ID とその保存時刻の 2 つの属性を持ったテーブルを作成した．cron によって 1 時間に 1 回の間隔で，7 日以上たった ID を消去するプログラムを動作させるようにした．

3.6.4 Example of use

MUA でのメールの送信から，最終的なあて先のユーザボックスへの配信までの，メッセージのヘッダの内容の変化を Fig. 3.8 に示した．

以下流れに沿って説明する．まず始めに，ユーザは Fig. 3.8(a) のような “X-HSRS:” フィールドを付けたメールを作り，SMTP-AUTH を用いてメールを送信する．

メールを受け取った送信側の MTA は，Fig. 3.8(b) のように “X-HSID:” フィールドをつけ，受信側 MTA に転送する．Fig. 3.6 における qmail-smtpd と qmail-queue でそれぞれ “Received:” フィールドがつけられるため，“X-HSID:” フィールドは，その 2 つの間に挟まる形になる．

受信側の MTA で送信側に問い合わせ，送信元を認証できたら，Fig. 3.8(c) のように “X-HSResult:” フィールドが付けられ，ユーザのメールボックスに配信される．“X-HSResult:” フィールドも 2 つの “Received:” フィールドの間に挟まる．

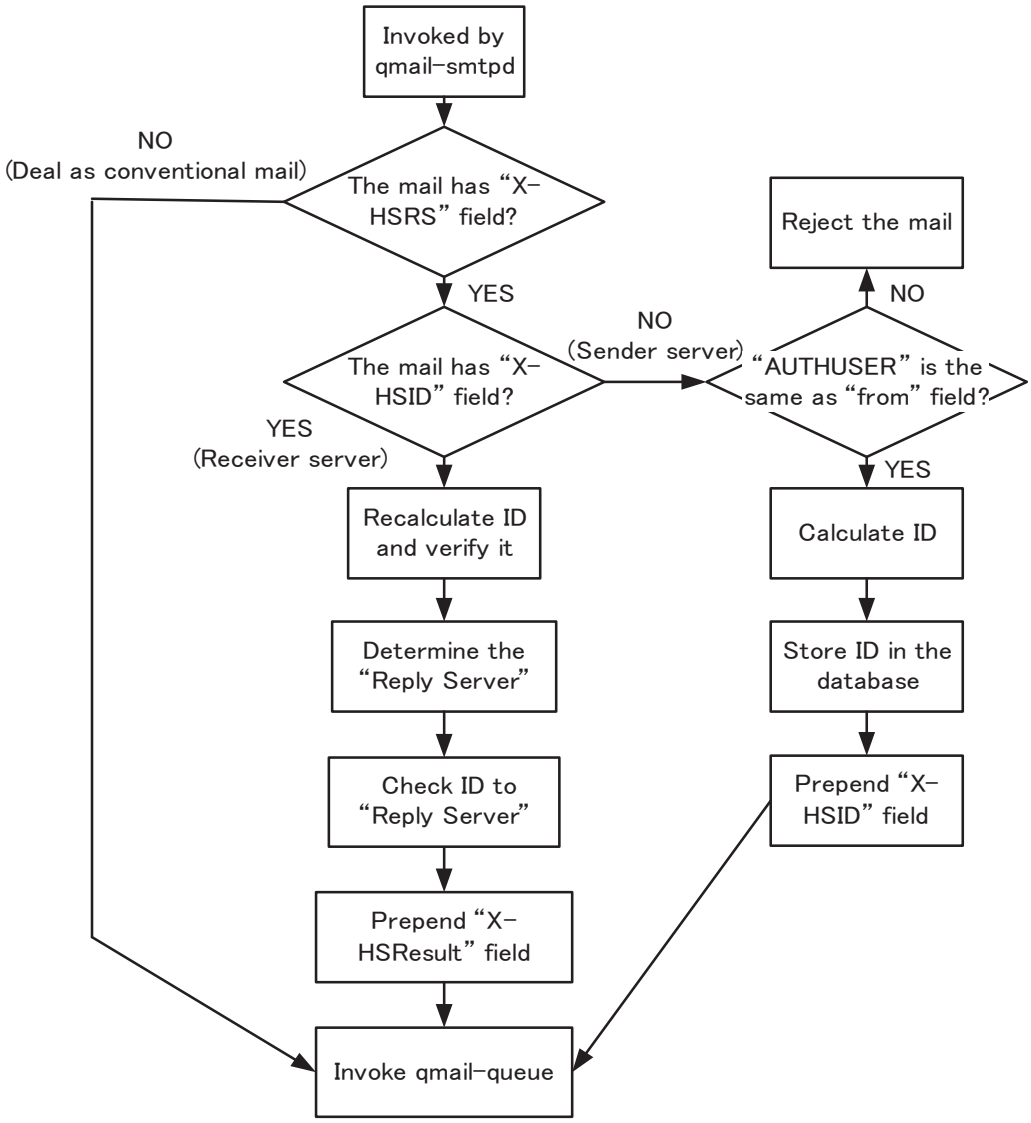


Fig. 3.7: Process flowchart of qmail-queue wrapper.

3.7　考察

本提案手法は，ドメインレベルでの送信元認証の 1 つと位置づけることができる．そのため電子署名と比較した利点は，ユーザ側からみると，MUA のソフトウェアを変える必要がなく，従来と同様のメールの送受信において，MTA で送信元認証が行われているという点である．メールの受信者も “X-HSResult” ヘッダだけ見られればよく，一般的な MUA であれば特別なプラグインソフトなどを用いることなく，送信元が認証されたかどうか確認することが出来る．

また，送信側の MTA と受信側の MTA の片方，もしくは両方が本システムに対応していない場合は，従来どおりのメールとして処理される．すなわち，送信側 MTA が本方式に対応していない場合はヘッダに何も記載されていないため，従来どおりのメールとして受信され，受信側 MTA が本方式に対応していない場合も，“X-HS***:” フィールドは理解不能なヘッダなため無視されてそのまま受信される．

他のドメインレベルでの送信元認証と比較すると，本提案手法の利点は，まず送信側 MTA が SMTP-AUTH を実装し，送信元アドレスのユーザ名の認証を行うことによって，受信側はユーザ名を含めて認証できるということである．これは送信側にとってみると，ある端末がウイルス等に感染して SPAM メールを送信しようとしても，送信元アドレスをユーザ名も偽装することが出来なくなるため，感染経路の特定がしやすいという利点がある．送信元アドレスは “From:” フィールドとしたため，“X-HSResult: Success” ヘッダがついていたものが SPAM メールだった場合は，送信側のドメイン全体が SPAM 業者の運営するものでない限りは，そのメールアドレスの持ち主が SPAM メールを送信していることになる．したがってそのメールアドレスを第三者機関に知らせることで，SPAM メール送信者リストを作成しやすい．SPF や，Sender-ID のように送信元アドレスがどれであるか探す必要はない．

SPF と Sender ID に対して優れている点は，転送されてきたメールや，ML から送られてきたメールにおいて，途中のサーバが本システムに対応していなくても，一番大元の送信元ドメインに対して確認することが可能であるという点である．これは SPF や Sender ID において，SPAM 業者が SPAM メールをあたかも他から転送されて来たかのごとく見せかけて送った場合，受信側で送信元認証を行うことができないことに比べ，優れている点であると言える．

また，DomainKeys のような公開鍵暗号を使う手法に比べると，鍵を生成したり管理したりする手間を省くことができる．DomainKeys はメール全体に対して署名をするため，改行コードなどによりエラーが起こりやすい可能性があるが，本提案手法は，4 つのヘッダのみを ID の要素として採用しているため，よりエラーが起こりにくいと考えられる．

本システムは，メールを送信するユーザがメールに特殊なヘッダをつけた時のみ，送信元認証の仕組みが働くようにした．これは，例えば企業が顧客にメールを出す時に，“このメールだけは重要なメールなので受信側で SPAM フィルタにひっかかってほしくない” と思ったメールに対し，“X-HSRS” ヘッダをつけ，受信側に送信元を確認させるなどといった使い方を考えたためである．そのため，SPAM メールというよりはフィッシング詐欺メールの防止に効果があるシステムであると言える．MUA によってはメールに拡張ヘッダをつけられないソフトも存在するため，送信元を偽装した SPAM メールを完全に食い止めるためには，サーバ側で全てのメールに ID をつけ，全てのメールの問い合わせに答えるようにしなければならないと考えている．

本提案手法の欠点は，従来の SMTP に比べて，ヘッダを解析しなければならないことと，問い合わせのコネクションを張ることから，処理にかかるオーバーヘッドが増加するという欠点がある．また悪意のあるユーザからの攻撃としては，送信元を偽ったメールをたくさん投げることによる問い合わせサーバに対する DoS 攻撃が考えられる．そして，正規のメールの送信を覗き見することで，ID を構成する要素を全て真似たメールの再送攻撃を行うことも可能である．

3.8　まとめと今後の課題

本章では，送信元認証の 1 方式として，送信側のサーバが送ったメールの情報を保持しておき，受信側のメールサーバは送信側のメールサーバに対し，メールを送ったことを問い合わせて確認してから受信するメール送信システムを提案した．本提案手法により，メールの受信者は送信元アドレスを転送されてきたメールであってもユーザ名まで含めて認証することが可能である．

今後の課題を以下に挙げる．

- 現在結果の表示は “X-HSResult” フィールドに Sucess か Error かの二種類のステータスしか定めていない．これではエラーの時に何が原因でエラーになったのか受信者にわからない．そ

もそも Reply Server が存在しなかったのか，ID が存在しなかったのか，通信中にエラーが発生したのかなどそれぞれのもっと詳しい状況を定義する必要がある．

- 本論文では処理速度向上などには一切取り組まなかった．メール一通あたりの処理時間のオーバヘッドは，Pentium4 1.7[GHz] の WindowsPC での VMWare 上の Linux サーバにおいて，おおよそ 0.01 から 0.02 秒ほどオーバーヘッドがかかっていた．主にデータベースでのアクセスに時間がかかっていたため，この速度を上げるための工夫も必要であろう．

```
X-HSRS: mail-a
Date: Wen, 6 Oct 2004 15:35:55 +0900
To: bob@example.net
From: alice@example.com
Subject: Test mail

This is a test mail.
--
alice@example.com
```

(a) The user composes an email.

```
Received: (qmail 10973 invoked from network); 6 Oct 2004 15:36:35 +0900
X-HSID: b7bd84ee13d89f05dcc30e5f9ec471e9
Received: from AlicePC.example.com (192.168.1.1)
by 172.20.1.1 with SMTP; 6 Oct 2004 15:36:20 +0900
X-HSRS: mail-a
Date: Wen, 6 Oct 2004 15:35:55 +0900
To: bob@example.net
From: alice@example.com
Subject: Test mail

This is a test mail.
--
alice@example.com
```

(b) The email is signed.

```
Return-Path: <alice@example.com>
Delivered-To: bob@example.net
Received: (qmail 4800 invoked from network); 6 Oct 2004 15:37:33 +0900
X-HSResult: Success
Received: from mail-a.example.com (172.20.1.1)
by mail-b.example.net with SMTP; 6 Oct 2004 15:37:22 +0900
Received: (qmail 10973 invoked from network); 6 Oct 2004 15:36:35 +0900
X-HSID: b7bd84ee13d89f05dcc30e5f9ec471e9
Received: from AlicePC.example.com (192.168.1.1)
by 172.20.1.1 with SMTP; 6 Oct 2004 15:36:20 +0900
X-HSRS: mail-a
Date: Wen, 6 Oct 2004 15:35:55 +0900
To: bob@example.net
From: alice@example.com
Subject: Test mail

This is a test mail.
--
alice@example.com
```

(c) After successful verification.

Fig. 3.8: Example of use

第4章

アンケート返信メール
自動集計機能の提案

4.1　はじめに

第 3 章では，送信元認証の 1 方式として，送信側のサーバが送ったメールの情報を保持しておき，受信側のメールサーバは送信側のメールサーバに対し，メールを送ったことを問い合わせて確認してから受信するメールシステムを提案した．具体的には，各メールごとに ID を発生させ，それをヘッダ内に挿入することで実現した．

これは，増え続けていく SPAM メールやフィッシング詐欺に対して，ヘッダ内に ID を挿入することにより，メールの利便性や信頼性が，“これ以上悪くならないようにする” 機能として位置づけることが出来る．

本章では，送信側のサーバで ID を発生させ，それをヘッダに挿入すると共に，その情報を保持しておくことで，より電子メールを便利に使えるような付加機能の実現について検討する．

4.2　動機

電子メールはもはや，人と人とのコミュニケーション手段として，欠かすことのできないアプリケーションとなった．これは，インターネットが世の中に広く普及したことと，メールが CSCW[40] における非同期分散型のアプリケーションとして，同じ場所，同じ空間にいない人とでも手軽にコミュニケーションが取れることが大きな要因であろう．しかしメールを用いると不便なことも の 1 つとして，多人数に対してアンケートを取る場合が考えられる．

第 1 章でも述べたように，メールは単方向性のアプリケーションである．この単純な機能を補うため，以下の 2 つの機能はすでに存在する．

1. エイリアスやメーリングリスト（以下，ML），メールマガジン（以下，MM）など，サーバでメールをコピーすることにより多人数に対して同じメールを送る機能．
2. ある人と双方向にコミュニケーションのやりとりをするために，お互い届いたメールに対し，“返信” する機能

ML 上で議論したりする場合は，上記の 1. と 2. の機能を組み合わせて使っている．

ここで今述べたアンケートを取る場合を考えてみる．MUA の宛先欄に複数のメールアドレスを記入したり，上記の 1. を利用すれば，同じメールを複数の人に送信することができるため，アンケート取るためのメールを送ることは簡単である．しかしそれに 2. の機能を利用した返信メールは，複数のメールを束ねる機能がないため，1 通 1 通別々にアンケートの送信者に届けられるため，送信者がそれらを集計するするのはとても大変である．

メールは，“1 対 1” のコミュニケーションと “1 から多へ” のコミュニケーションの手段としては優れているが，アンケートのような “1 から多” そして “多から 1” といった形式のコミュニケーション手段には，向いていない．

ML によっては，そこに投稿されたメールを 1 通届くたびに配信するのではなく，一定間隔でその間に届いたメールをまとめて配信する “ダイジェスト” という機能はあるのものの，それはメールをただコピー&ペーストしただけの機能であり，集計されているとは言い難い．そしてそれらはあくまで ML の機能なので，その場限りの特定のメンバには適用できず，最初から配信するグループを決めておかなければならない．

現在インターネットで企業などがアンケートを取る際には，主に Web が用いられている．まず入力フォームを用意し，アンケートの対象者に URL をメールで知らせ，そこにアクセスして入

力してもらう．そして結果はまとめて Web 上に表示させたり，1 通のメールとしてアンケートを取った人にメールを返すといった手法が多い [41].

しかし Web を用いることの問題点は，CGI を使える Web サーバをを利用しなければならず，一般のユーザ[1]が気軽に使えないことである．また返信する側も，URL を知らせる手段にメールが使われているのに，わざわざブラウザを開いて作業をしなければならず，面倒である．アンケートを取られる人は，送られてきたメールに返信するだけで回答でき，アンケートを取った人は，その結果を 1 通のメールとして受け取ることができる，といったことを，ML ような現在多くの人が利用するメールを拡張した機能の 1 つとして実現したい．

以下本章においては，アンケートを取る人を Alice，アンケートを取られる人を Bob 達として話を進め，アンケートを取ることが目的のメールを “アンケートメール”，それに対する返信のメールを単に “返信メール”，それ以外の従来どおりのメールを “普通のメール” と呼ぶことにする．

4.3　目的

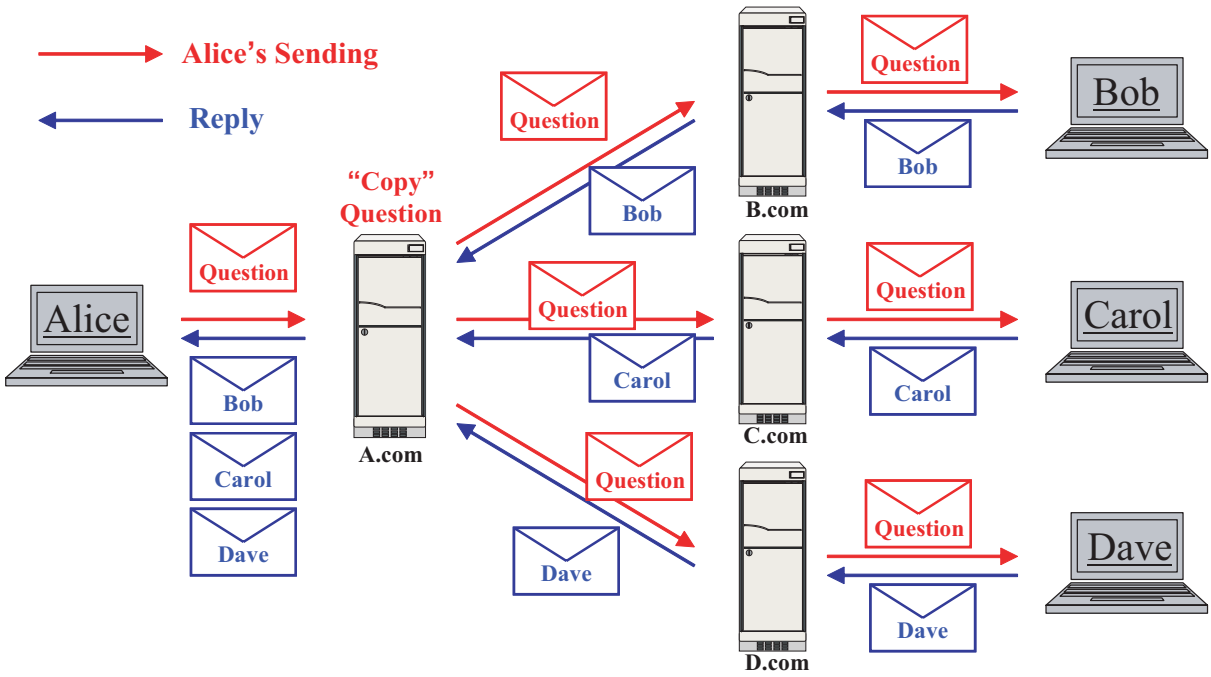


Fig. 4.1: Alice must view reply mail separately.

本論文では，電子メールの 1 つの機能として，一般のユーザがアドホックなメンバに対し，気軽にアンケートを取ることができるシステムについて検討する．

現在の電子メールでは，Fig. 4.1 のように，Alice は Bob 達 3 人から返信メールを 1 通ずつ見なければならない．本提案手法では Alice が送ったメールを Bob 達が返信し，その返信メールは自動的に集計して 1 つのテーブル（表）にし，Alice に 1 通のメールとして返すということを，Alice にも Bob 達にもできるだけ負担をかけずに行いたい．

[1] “一般のユーザ” とは，コンピュータに関してサーバ管理者のような深い知識は持っておらず，一般に常識的な知識を持っているユーザのことを指す．

第 3 章で，“送信側の MTA” が送信メールの情報をデータベースに保存し，後に利用するということについて検討したので，これを拡張することにした．メールサーバがアンケートメールをそのドメイン内のユーザの MUA から受け取ると，集計用のテーブルをデータベースに作成し，メールを転送する．そして，各返信メールの内容をテーブルに保存し，一括してユーザに返すというシステムにする．第 3 章でいうところの “受信側の MTA” は従来どおりのメール転送とメール配信しか行わないため，このメールサーバが提供するものは，メーリングリストのようなドメイン外のユーザでも利用できるような機能ではなく，自分のドメイン内のユーザに対する機能として動作させることにする．Fig. 4.1 の例で述べると，Alice がアンケートメールを送る場合には，A.com 内のメールサーバ上でメールを集計する．もし仮に Bob がアンケートメールを送りたい場合は，B.com 内のメールサーバ上でメールを集計するということになる．尚，A.com 内のメンバは，第 3 章と同様，拡張ヘッダをつけてメールを送信できることを前提とする．

本システムは第 3 章のシステムと異なり，“受信側の MTA” は特別な処理はしない．本章で以下 “MTA” と言った場合は “送信側の MTA” を指し，“メールサーバ” と言った場合はその MTA が動作しているサーバのことを指すこととする．

返信メールの一括化を行うには，アンケートメールごとに ID を割り振る必要がある．それには，第 3 章で用いた ID をそのまま使用することにした．そして，その ID を第 3 章と同様，ヘッダの中に埋め込み，返信メールのヘッダ内に挿入してもらうことで，どのアンケートメールに対する返信メールかを特定することにした．

4.4　設計の基本方針

4.4.1　普通のメールとの区別

メールサーバが，アンケートメールや返信メールを普通のメールと区別するためには，何らかの仕組みが必要であり，以下の 2 つのオプションが考えられる．

専用のメールアドレスを用いる

ML のように，アンケートメールや返信メールを処理するための専用のメールアドレスを作成する．そして，そのアドレスに送られてきたメールはアンケート集計に関するメールとみなし，返信メールを集計するプログラム（以下，このプログラムのことを “集計用プログラム” と呼ぶ）に渡すことによって，処理を行う．

拡張ヘッダを用いる

第 3 章のように，ユーザが拡張ヘッダをつけて送信する．MTA がヘッダを解析し，その拡張ヘッダが存在した場合に，アンケート集計に関するメールとみなして特別な処理を行う．

したがってこの 2 つの組み合わせによって Table 4.1 に挙げる A から C の 3 つの手法が考えられる．

Table4.1: Questionnaire mail distinction.

専用のメールアドレスを用いる	拡張ヘッダを用いる	A
	拡張ヘッダを用いない	B
専用のメールアドレスを用いない	拡張ヘッダを用いる	C
	拡張ヘッダを用いない	区別できない

4.4.2　アンケートメールの送信と返信手法

アンケートメールの送信

アンケートメールの送信においては，本論文では Table 4.1 の C を採用することにした．理由は以下の通りである．

専用のメールアドレスを用いると，実際のアンケートの送信先をどこか他の場所に書かなくてはならない．その場合は，

- “Subject:” フィールドに書く
- メールボディの先頭に書く
- 拡張ヘッダを使い，その中に書く

という 3 つの場所が考えられる．“Subject:” フィールドに送信先を書いてしまうと，Alice は “Subject:” フィールドにはそのアンケートメールの用件について書けなくなるため，Bob 達に用件を伝えにくい．メールボディの先頭に書く場合は，複数のメールアドレスを羅列する場合もあるため，どこまでがアンケートの送信先なのかを指定する必要がある．またボディの一行が長いことにより途中で勝手に改行されたりすることで，エラーが起こりやすい．拡張ヘッダを用いる場合も同様である．さらに，MUA によってはアドレス帳で送信先の名前をクリックしたりするだけで，“To:” フィールドや “Cc:に” フィールドにアドレスが挿入されるものが多いが，そういった機能も使えずに，1 つ 1 つコピー&ペーストしなくてはいけなくなる．Alice にとって，わかりやすく使いやすいシステムということを考えると，アンケートの送信先（Bob 達のアドレス）は普通のメール “To:” フィールドや，“Cc:”，“Bcc:” フィールドに書くことであろう．

以上の議論から専用のメールアドレスを用いずに，拡張ヘッダを用いて送信するのが最善と考えた．

アンケートメールへの返信

逆にアンケートの返信に関してはどうだろうか．アンケートを取る状況を考えると，Bob 達は携帯電話のメールアドレスを使用していることも想定したい．そのため，Bob 達は返信メールに拡張ヘッダを付けることができない可能性がある．そこで本論文では Table 4.1 の B を採用することにした．この集計用に用いる専用のメールアドレスを，以下 “集計用メールアドレス” と呼ぶことにする．

アンケートメールの送信元は Alice なのに対し，その返信先を集計用メールアドレスにするためには，“Reply-To:” フィールドが利用できる．一般の MUA は，設定にもよるが，“返信ボタンを押す” と，“Reply-To:” フィールドのメールアドレスが送信先に設定される．Alice の負担を軽減するため，MTA が返信先として “Reply-To:” フィールドをつけることにする．

4.4.3　ID の挿入

第 4.3 節でも述べたが，どのアンケートメールに対する返信メールかを特定するために，アンケートメールを受け取った MTA は，ヘッダ内に ID を挿入して Bob 達に転送し，Bob 達は返信メールでその ID を返す必要がある．しかし，一般の MUA ではメールに返信をした場合，そのメールについていたヘッダの内容は一切引用されず，新しくヘッダを付け直してしまう．ID を返すための 1 つの方法として，Bob 達がアンケートメールのメッセージの中から ID を抜き出して，“Subject:” などに手動で入力することが考えられるが，これは Bob 達にとって大変煩わしく入力ミスが起きる可能性が高い．Bob 達にとって一番簡単な方法は，“返信ボタンを押す” という作業のみで，ID を返せるようにすることである．したがって本提案手法においては，“Reply-To:” フィールドに書き込む集計用メールアドレスの中に ID も埋め込むことにする．

4.4.4　本提案手法の前提条件と概要

これまでの議論によって，本提案手法を利用するためには，以下のような条件が必要である．

- Alice は，一般的な PC の MUA を用いていて，拡張ヘッダを使用できるとする．
- Bob，Carol，Dave は，PC の MUA だけでなく，拡張ヘッダが使用できない携帯電話のメールを使用している場合もあるとするが，受信したメールの “Reply-To:” フィールドを参照でき，そこに書かれているメールアドレスに返信できるものとする．

これは現状の電子メールの環境を考えても，十分にあり得ることである．

設計の概要を Fig. 4.2 に示す．尚，Fig. 4.2 においては，B.com のメールサーバは省略している．Alice は，メールに拡張ヘッダをつけて，直接 Bob 達に送信する．返信メールは “Reply-To:” の集計用メールアドレスに向けて送信する．集計結果はデータベースに溜めておき，返信期限を過ぎたら Alice に一通のメールとして返す．

4.5　要求される機能

本節では，Fig. 4.2 において，ユーザに具体的にどのような操作をさせる必要があるか，それによってどのような機能が利用できるのかについて検討する．

4.5.1　アンケートメール送信者の操作

アンケートの種類や内容は状況によって様々である．Alice(アンケートメール送信者) としては，できるだけその状況応じたアンケートの取り方をしたい．例えば，

> 3 個の項目について，15 人にアンケートを取るが，とりあえず 5 人分返信があったところで一旦途中経過を（メールで）返してほしい．その後は，1 日に 1 回午後 3 時に途中結果のメールをみたく，さらに最終結果は 5 日後の午後 8 時に欲しい．それ以降の返信メールは alice-reply-late@A.com に転送して欲しい．

といった複雑な指定をしたい場合もあるが，

> 3 個の項目について，5 日後までにアンケートを取りたい．

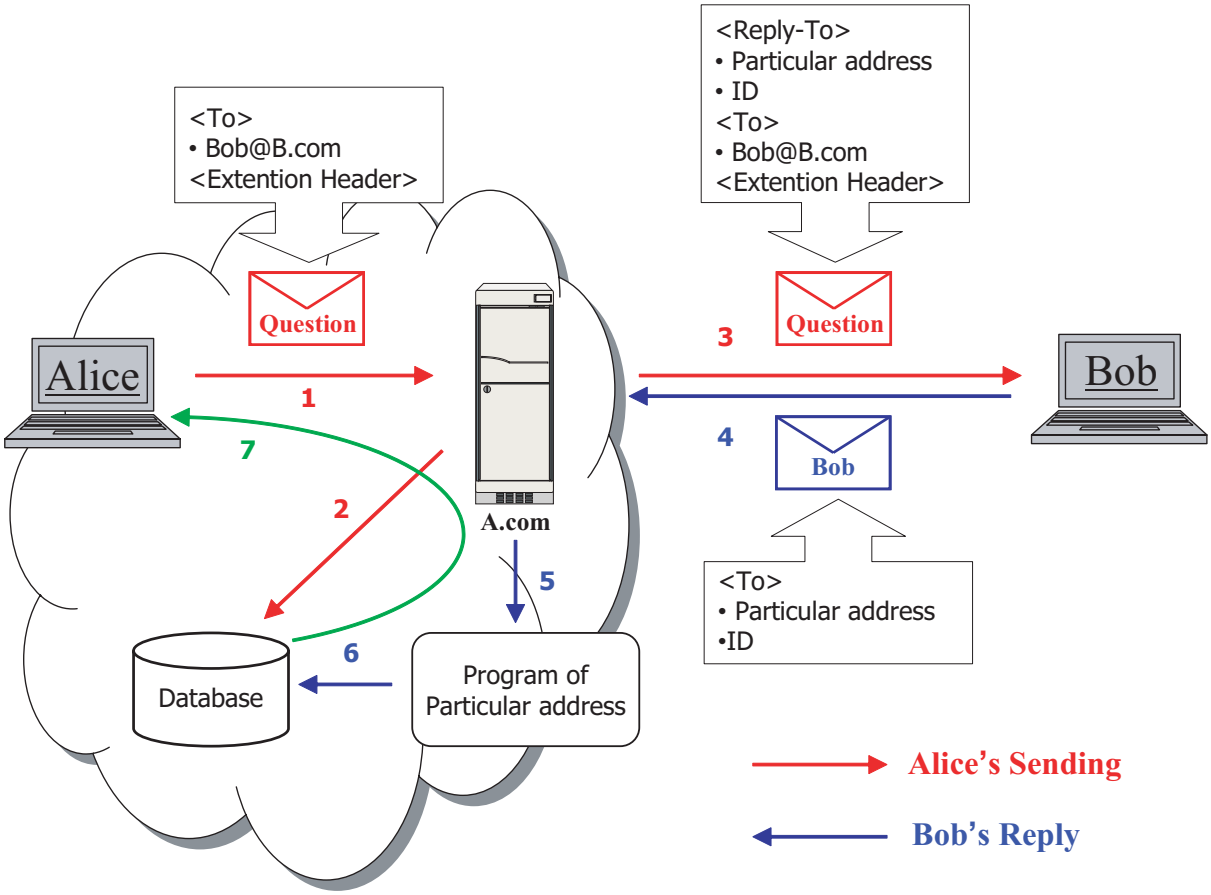


Fig. 4.2: Overview System of Automatic Tabulation.

といった単純な場合もある．

アンケートの特性ごとに，それに合わせたアンケートの取り方が出来るということはそれだけ高機能であると言えるが，逆に様々なことを指定しなければならないので，使い勝手が悪く，一般人にとって敷居の高いものになってしまう．そのため，Alice が指定することは，全てのアンケートメールに共通して最低限 MTA に知らせなければならないこと，すなわち「返信期限」と「質問項目数」の 2 つとする．

4.5.2　返信メール用フォーマットの作成

返信メールは，集計用プログラムが解析しやすいように，決まった形式になっている必要がある．ただ Alice が自分でフォーマットなどを作成することは，大変煩わしく，予めそのフォーマットを覚えていなければならないという問題がある．質問項目数を提示するだけで済むようにしたい．また Bob 達にとってみると，予めフォーマットがメッセージのボディに記入されていて，その本文を引用して回答を書き込むだけで返信できるようにしたい．したがって，MTA でメッセージのボディの頭に質問項目数に応じた返信用のフォーマットをつけて送信することにする．

4.5.3　集計結果メールとその形式

アンケートは返信期限の過ぎた場合，例え誰かが返信してなくても，Alice に一通の集計結果メールとして返す．それ以降に届いた返信メールに対しては，“もう返信期限が過ぎました” というエラーメールを返信者に返し，アンケートの送信者に直接返信してもらうように促す．

本提案手法においては，集計結果は CSV 形式の添付ファイルの形式で Alice に返すことにした．CSV 形式は主に表計算ソフトやデータベースソフトがデータを保存するときに使う形式であるが，汎用性が高く，多くの電子手帳やワープロソフトなどでも利用できるため，異なる種類のアプリケーションソフト間のデータ交換に使われることも多いため，妥当であると考えた．例えば Microsoft®Windows®では，Microsoft®Excel®に.csv ファイルを関連付けていれば，Fig. 4.3 のように簡単に結果を表にして見ることができる．また当然一般のテキストエディタでも開くことが出来る．

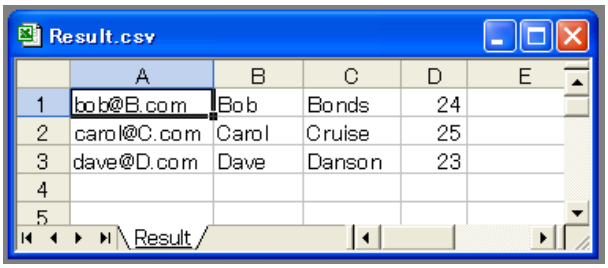


Fig. 4.3: View the CSV data.

4.5.4　途中結果の閲覧

以上の仕様で最低限実現に必要な機能は満たすことができる．もう 1 つ，いつでも現在の返信状況を参照できるようにする機能が欲しい．その場合は，返信メールの “Subject:” フィールドに特別な文字を入れると，利用できるようにする．本提案手法においては，各アンケートは全て ID で管理し，その ID はサーバ上で計算されるため，この機能を使うには自分にもメールを送信しなければならない．

4.6　実装

本提案手法を，第 3 章と同様，MTA は qmail を，データベースは PostgreSQL を用いて実装した．以下，現在の実装の詳細について，小節に分けながら説明する．

4.6.1　拡張ヘッダ

本実装においては，アンケートメールを送りたい場合には，“X-QUERY:” というフィールドを付けて送信することとした．MTA が自分のドメインから来たメールと認識しなければならないため，このフィールドを用いる場合は，第 3 章で述べた “X-HSRS:” フィールドを併用し，SMTP-AUTH を用いてメールを送信しなければならない．もし “X-QUERY:” が存在するが “X-HSRS:” フィールドが存在しなかった場合は，DATA コマンドの応答として，

```
554 X-QUERY must be put with X-HSRS
```

というエラーを返し，メッセージを受け付けない．

“X-QUERY:” フィールドの中身は，n=の後に質問項目数を，d=の後に現在から返信期限までの日数をそれぞれ続けることにより指定する．

```
X-QUERY: n=3, d=5
```

3 項目についてアンケートを取り，5 日後にサーバから集計結果を返信，という意味になる[2]．もし “X-QUERY:” のフォーマットが間違っていた場合は，DATA コマンドの応答として，

```
554 X-QUERY format error
```

というエラーを返し，メッセージを受け付けない．

4.6.2　メッセージボディの様子

Alice が上記のヘッダをつけて送った場合，MTA はメッセージボディの頭にアンケート用の以下のようなフォーマットを付けて転送する．

```
--Ans. Begin--
A1 =
A2 =
A3 =
---Ans. End---
```

n=4 の場合は，A4 = までつくことになる．したがって，Alice は 3 つの質問項目がある場合は，“X-QUERY:” フィールドに n=3 と書いた上で，メールボディに

A1 には姓を，A2 には名を，A3 には年齢を書いてください’

といったように各返信項目を書いておく必要がある．

Bob 達は以下のように記入して返信するとする．

```
> --Ans. Begin--
> A1 = Bob
> A2 = Bonds
> A3 = 24
> ---Ans. End---
```

--Ans. Begin--と---Ans. End---の間にはさまれた A<x> = に続く文字列が，それぞれの項目ということになる．---Ans. End---以下の文章は無視されるため，Bob 達は “返信ボタン” で本文を引用し，各 A<x> = に回答するだけでよい．また返信の引用符は「>」でなくても構わない．

[2]質問項目数と返信期限の順番を逆に書いても正常に動作する

4.6.3　データベースのフォーマット

データベースのフォーマットを Fig. 4.4 に示す.
送信されたアンケートメールの一覧として，以下の 5 つの属性を持ったテーブルを用意した.

> ID，アンケートメールの送信元アドレス（集計結果返信先のアドレス），アンケートが送信された時刻，返信期限，質問項目数

これをテーブル A と呼ぶ．現在の実装では，アンケートが送信された時刻は使われていないが，将来の拡張を考え用意した.
MTA では，“X-QUERY:” フィールドがついたメールを受け取った場合，テーブル A に INSERT 文を用いてデータを挿入する．また，その後にアンケート集計用テーブルを CREATE TABLE を使って作成する．そのテーブルの属性は以下のようになっている.

> 返信者のアドレス（Bob 達のアドレス），項目 (1)，項目 (2)，項目 (3)，…，項目 (質問項目数)

このテーブルは各アンケート（ID）ごとに 1 つ存在することになる．これらをテーブル B[ID] と呼ぶ.

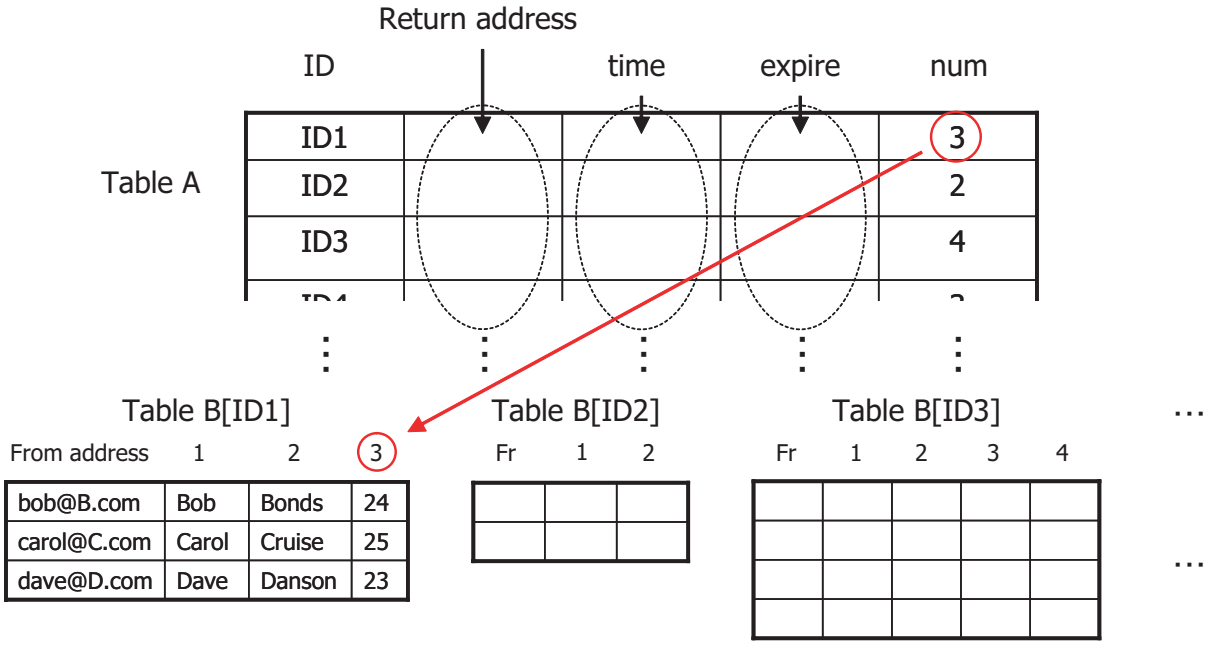


Fig. 4.4: Table Format.

4.6.4　MTA の処理の流れ

MTA は，アンケートメールの送信の時にのみ特別な動作をする．返信メールは一般の MTA と同様の動作をし，fml を改造した集計用プログラムに渡すだけである．Fig. 3.7 に少し余計な処理が加わった程度であるが，MTA の処理の流れを Fig. 4.5 に示す.

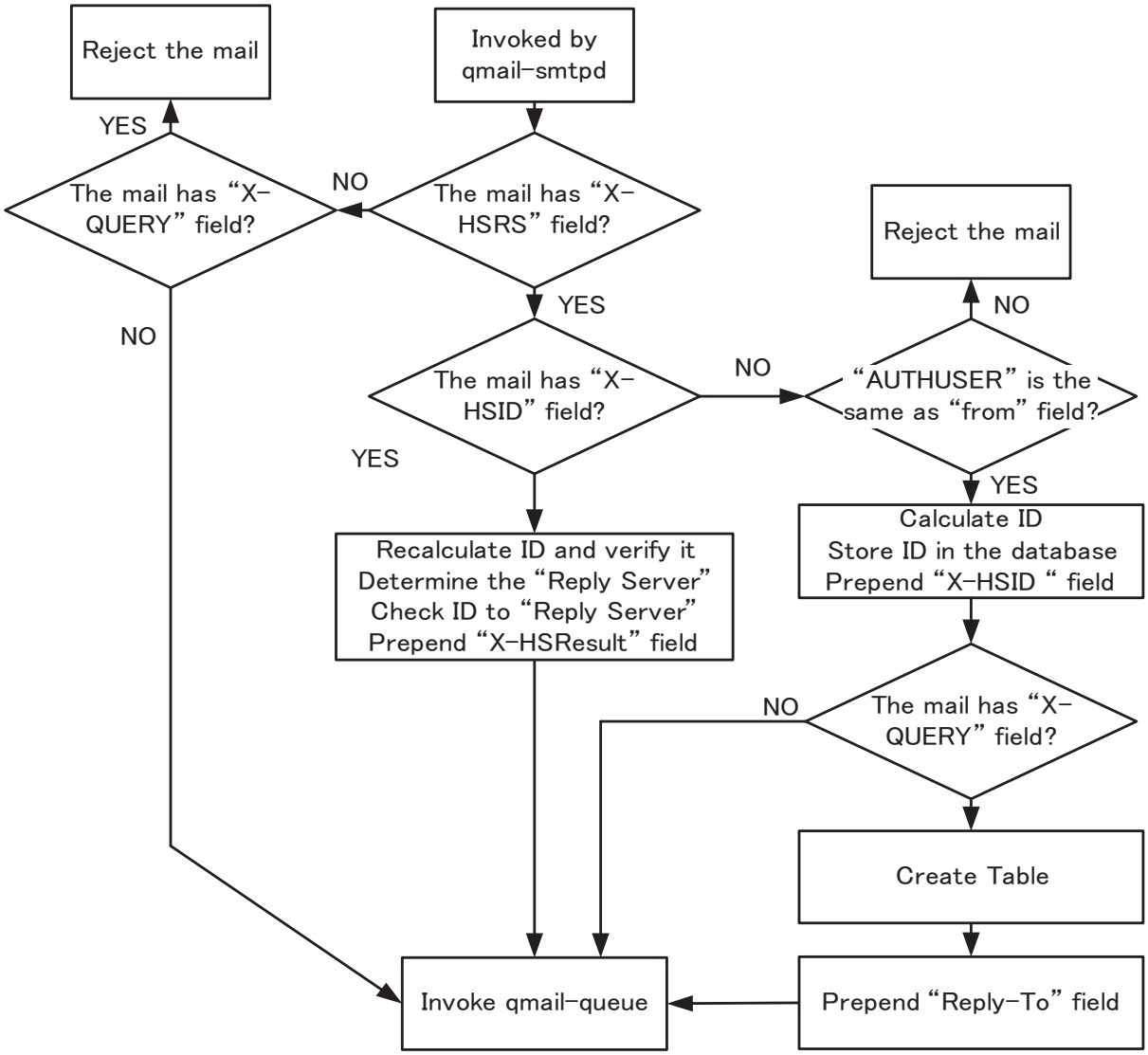


Fig. 4.5: Process flowchart of qmail-queue wrapper.

4.6.5　集計用メールアカウントと fml

本実装では，集計用メールアドレスのアカウントを reply とした．そして，返信メールの集計用プログラムとして，ヘッダの解析に fml のバージョン 4.0[42] を利用した．

fml とは，深町賢一氏が作成する，GNU General Public License に基づく MLM(メーリングリストマネージャ) である．言語として Perl が採用されており，Perl が動作するほとんどの環境で動作させることができる．実際に，UNIX 系のさまざまな OS はもちろん，Windows NT 4. や Mac OS X Server 環境などでも動作確認が進んでおり，ユーザの好みの環境で動かすことが可能である．

まず fml を「fml」という専用のユーザを作ってインストールし，fml ユーザで，

```
$ makefml newml reply
```

することによって reply という ML を作った．そして，その config.ph の$PERMIT_POST_FROM

を anyone とすることで誰からのメールでも受け付けるように設定し，$DO_NOTHING 変数を 1 にセットして配信処理とコマンド処理をスキップさせることで，fml をヘッダの解析のみに利用するようにした．

4.6.6　集計用メールアドレスへの ID の埋め込み

第 4.4.3 節で述べたように，MTA は集計用メールアドレスの中に ID も埋め込む．よって “Reply-To:” フィールドの形式は，以下のようにした．

```
Reply-To: reply-<ID>@A.com （A.com は MTA のドメイン名）
```

ただし，reply-<ID>というメールアドレスは，全て reply というアカウントで受け取る必要がある．このためには，qmail の拡張アドレスという機能が利用できる [37]．拡張アドレス機能を使うと，ユーザ (reply) は自分のアカウントである reply だけでなく，reply-*anything* という形式のメールアドレスも管理することができる．

まず，以下のようにファイル/var/qmail/users/assign のワイルドカード型割り付けにより，メールアドレスとアカウントの対応づけを行う．

```
# vi /var/qmail/users/assign
+reply:fml:512:512:/var/spool/ml/etc/qmail/alias:-:reply:
# /var/qmail/bin/qmail-newu （変更の反映）
```

assign は qmail-newu で処理された後，Fig. 3.6 の qmail-lspawn により使われ，qmail-local のローカル配送を制御する．これにより，reply-で始まる任意のアドレスは，uid と gid が 512(fml の uid と gid) のアカウント名 fml に配送され，reply-*anything* へのメール配送は /var/spool/ml/etc/qmail/alias/reply/.qmail-reply-*anything* により決められる．

qmail-local は，reply-something へのメール配送時に，.qmail-reply-something が存在しなかった場合，qmail-local は，.qmail-reply-default を探す．そのため，以下のように記述しておくことで，reply-<ID>宛てのメールは，すべて fml.pl という fml を改造した集計用プログラムに渡されることになる．

```
# vi /var/spool/ml/etc/qmail/alias/.qmail-reply-default
|/usr/local/fml/fml.pl /var/spool/ml/reply
```

4.6.7　集計用プログラムの処理

fml にはフックという実行過程の中で fml 内部からダイナミックに呼ばれる関数が存在し，fml の動作を動的に変更したり，fml にはない機能を実現することができる [43]．本実装では，$START_HOOK フックを用いて返信メールを集計する処理を書いた．

Perl から PostgreSQL へのアクセスには，pgsql_perl5 という Perl モジュールを使用した．pgsql_perl5 インタフェースは，C 言語のインタフェースに非常に良く似ているため，C 言語からの移植が容易である．

4.6.8　集計結果の送信

cron によって 1 時間に 1 回の間隔で，集計結果を送信するプログラムを動作させるようにした．このプログラムでは，テーブルを検索して返信期限の過ぎた ID を見つける．その各 ID について，

テーブル B[ID] の内容を検索し，CSV 形式にする．それを MIME による添付ファイルにし，アンケートメールの送信元アドレスに送信する．

4.6.9　途中結果の閲覧

途中結果を閲覧したい場合には，“Subject:” フィールドに interim と書いて送ることとした．この場合は集計用プログラムは，返信メールではなく途中結果の閲覧要求メールとして処理し，その時点での集計結果を送信元に返信する．現段階の実装では，Alice は予めアンケートメールを自分自身にも送信してしていないと途中結果を閲覧することができない．また Alice だけでなく，Bob 達でも第三者でも途中結果が閲覧できる．

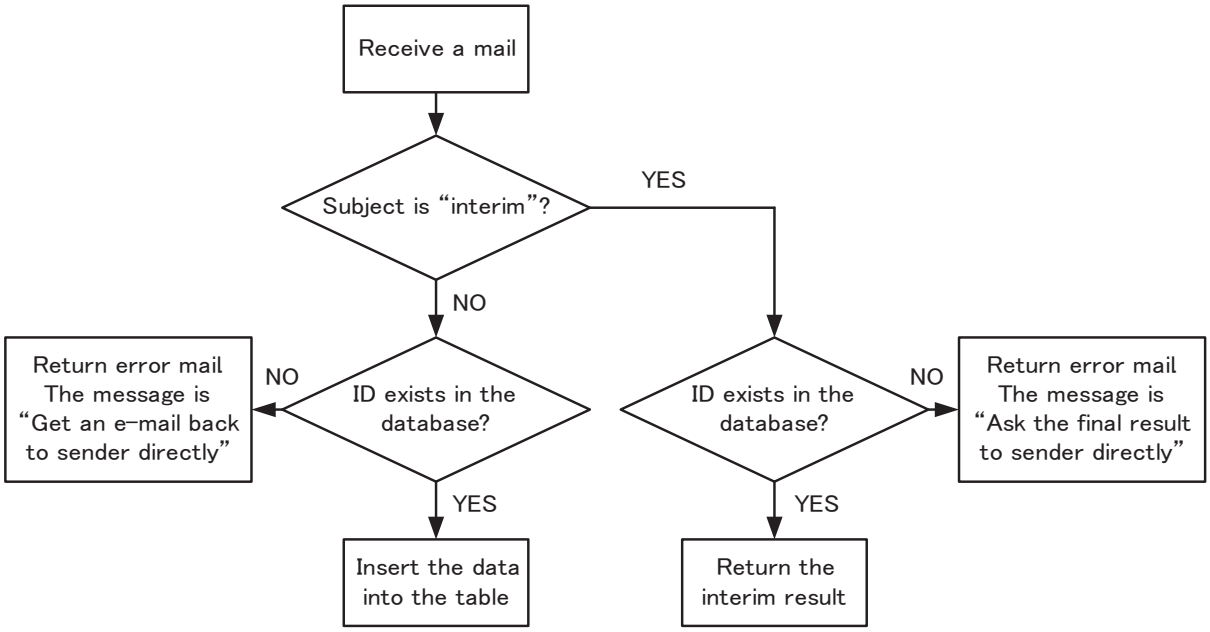


Fig. 4.6: Process flowchart of tabulation program.

集計用プログラムの処理を Fig. 4.6 に示す．メールを受け取ったら “Subject:” フィールドを調べ，返信メールであるのか，途中結果の閲覧要求メールであるのかを判断する．それぞれ “To:” フィールドのメールアドレスから ID を抜き出し，テーブル B[ID] にデータの挿入，または B[ID] の途中結果の送信を行う．もし返信期限が過ぎ，ID がデータベースに存在しなかった場合は，エラーメールを返す．返信メールの場合は，Alice に直接返事を出すように促す．途中結果の閲覧要求メールの場合は，最終的な結果は Alice に聞くように促す．

4.6.10　Example of use

本提案手法が仕様どおり動くことを確認した．実際のアンケートメールとヘッダの例を Fig. 4.7 に示す．Alice は，Fig. 4.7(a) のようなメールを作成する．それが MTA で “Reply-To:” フィールドと返信メール用フォーマットが付加され，Fig. 4.7(b) のような形になって Bob 達に送信される．Bob 達は，そのメールに対して “返信ボタンを押し”，回答を書き込むことで Fig. 4.7(c) のようなメールを返信する．

```
X-HSRS: mail-a
X-QUERY: n=3, d=5
Date: Wen, 6 Oct 2004 15:35:55 +0900
To: bob@B.com, carol@C.com, dave@D.com
From: alice@A.com
Subject: Question – about your name and age.

Reply to a question below quoting the message.
What is your first name(A1), last name(A2) and age(A3).
Alice.
```

(a) Alice composes an questionnaire email.

```
Reply-To: reply-c2356d1324f9238fea56a2435c0017ee@A.com
X-HSID: c2356d1324f9238fea56a2435c0017ee
X-HSRS: mail-a
X-QUERY: n=3, d=5
Date: Wen, 6 Oct 2004 15:35:55 +0900
To: bob@B.com, carol@C.com, dave@D.com
From: alice@A.com
Subject: Question – about your name and age.

--Ans. Begin--
A1 =
A2 =
A3 =
---Ans. End---

Reply to a question below quoting the message.
What is your first name(A1), last name(A2) and age(A3).
Alice.
```

(b) Bob, Carol and Dave receive the mail.

```
Date: Thu, 7 Oct 2004 18:12:34 +0900
To: reply-c2356d1324f9238fea56a2435c0017ee@A.com
From: bob@B.com
Subject: Re: Question – about your name and age.

> --Ans. Begin--
> A1 = Bob
> A2 = Bonds
> A3 = 24
> ---Ans. End---
>
> Reply to a question below quoting the message.
> What is your first name(A1), last name(A2) and age(A3).
> Alice.
```

(c) The reply mail(quote the text in the reply).

Fig. 4.7: Example of use

集計結果を実際にメールで受け取った様子を，Fig. 4.8 に示す．Fig. 4.8 において題名欄に見えている 3 件のメールは，それぞれ最終結果，途中結果，ID が存在しないことによるエラーメールとなっている．

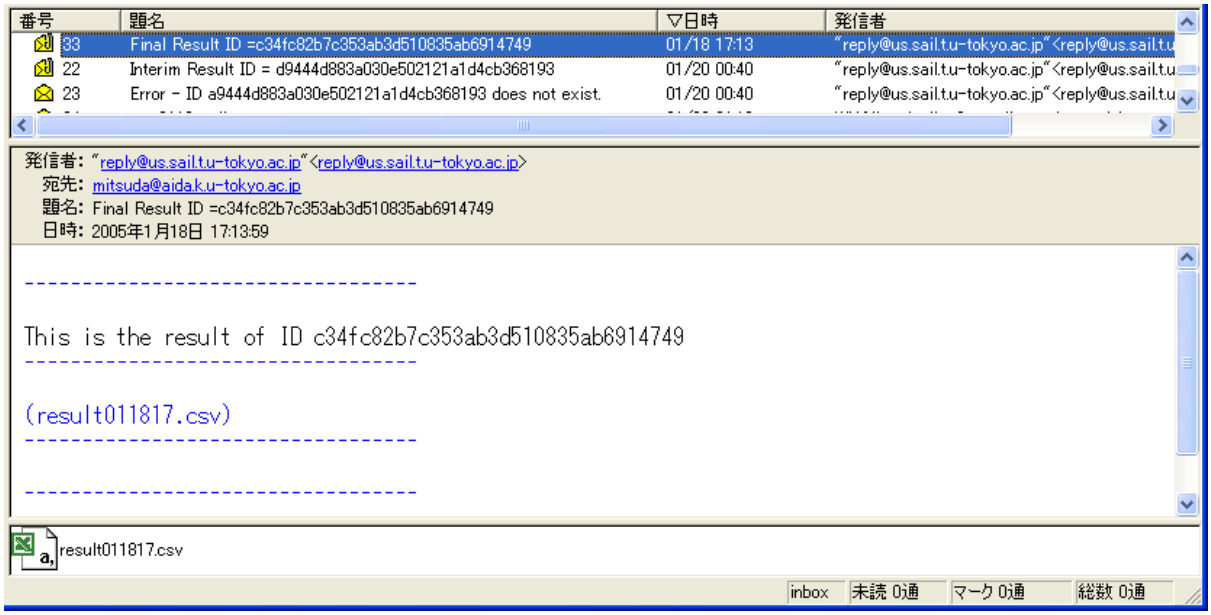


Fig. 4.8: Result of a questionnaire.

4.7　考察

本章では，アンケートメールの返信結果を集計して 1 つのメールとして送信者に返信する手法を提案した．本提案手法は，Web などで集計する手法と比較すると，Web サーバを立ち上げなくてもよく，返信する側もブラウザが必要がない．Alice にとっても Bob 達にとっても，MUA での必要最低限の操作で利用することが可能である．Alice は拡張ヘッダを付けるだけでアンケートメールを送ることができ，Bob 達もアンケートメールに対して，“返信ボタンを押し”，回答を書き込むだけで返信できる．

本提案手法は，拡張ヘッダを認識するために，MTA のプログラムに手を入れた．そのため，セキュリティを考慮し，ML のように他のドメインから来たメールに対してもサービスを提供することはせず，あくまで自ドメインの SMTP-AUTH の通ったユーザのみこの機能を提供することにした．ML の場合は，各 ML ごとにその目的に合わせた様々な設定をすること可能であるが，本提案手法はドメインごとに基本的な仕様を決め，残りの設定は全て Alice が行わなければならない．想定されるアンケートを取る状況に柔軟に対応することと，Alice にとって設定が複雑にならないように配慮する必要があった．

Alice にできるだけ負担をかけないように，拡張ヘッダでの設定項目は，返信期限と質問項目数の 2 つとしたが，これをもう少し増やし，設定しなかった場合のデフォルト値を決めることで，より複雑なアンケートの状況に対応できるシステムにする必要がある．例えば普通のアンケートは，不特定多数の人に取る場合を想定しておき，例えば `X-QUERY: n=3, d=5` として送信した場合は，途中結果は「Alice だけが見られる」ようにしておく．しかし，仲間同士でイベントの出欠

を調べたく，セキュリティをそれほど気にしない場合などは，途中結果は「誰でも見られる」ようにしたい．その場合は，`n=3`，`d=5`，`i=all` などのように 3 項目設定させることで実現する．別の項目として，同じ人から 2 度返信メールがあった場合は,「上書きしてしまう」か「別の人とみなす」かを決められたりするとよいであろう．このようにすることで，誰でも最低限の機能は気軽に使え，かつ詳しい人は，より高度な機能を利用できるシステムになる．

本提案手法はまだプロトタイプの実装段階であるため，いくつか改善すべき点がある．

Fig. 4.8 にあるように，集計結果メールは途中結果メールも含めて題名や本文には ID しか書いていないため，どのアンケートメールに対する結果であるかわかりにくい．集計結果メールの題名には，アンケートメールの題名の前に例えば【集計結果】などの文字ををつけたものを用い，本文にはアンケートメールの送信時刻なども入れることで，Alice にとって有益な情報を返信するするべきである．

本実装においては，集計結果は添付ファイルを開かなければ見ることができない．ボディにも集計結果を載せておけば，MUA でそのメールを選択するだけで，結果を閲覧することが可能になる．集計結果の内容には各返信メールの返信時刻を入れたりすることで，集計結果を後で活用しやすくする工夫も必要である．

またアンケートメールの本文に返信用のフォーマットと共に，返信先アドレスを書いておくことで，Bob 達が誤って Alice に直接返信してしまうことを防ぐようにしなければならない．

追加したい機能としては，Alice が途中結果を見て返答率が悪いと判断した時に，返信期限の延長をメールで要求できるようなものが挙げられる．

4.8　今後の課題

今後の課題としては，まず前節の考察を深め，より使いやすくより便利な仕様について検討する必要がある．最終的には実際に多くのユーザに使ってもらい，どれほど使いやすいかを評価しなければならない．

セキュリティについても考慮する必要がある．本提案手法では，アンケートの送信者に対しては，第 3.2 節で述べた SMTP-AUTH を用いたユーザ認証を用いているため，送信元の偽装は防止しているが，逆にアンケートの返信者は送信元を認証していないため，他の返信者を装うことができてしまう．そのため，本章のシステムを第 3 章のシステムと組み合わせ，返信メールに送信元認証を用いることについても検討したい．

第5章

結論

5.1　まとめ

5.1.1　電子メールの高機能化

本論文では，電子メールが単方向性のアプリケーションである点に着目し，その欠点を補うため，2 つのことに取り組んだ．

送信元認証におけるハンドシェーク活用の提案

SMTP には送信元の認証機能が存在しない．電子署名のように End-to − End で送信元認証をすることによってメッセージの完全性を保証する方法は存在するが，全ての一般ユーザが使うようになるにはまだ多くの時間がかかる．そこで，ドメインレベルで送信元認証を行う方式を提案した．現在研究されている SPF や Sender ID と異なり，転送されてきたメールなどに対しても，大元の送信元について認証することが可能である．SMTP-AUTH の技術と組み合わせることにより，2 段階に認証を行い，メールアドレスのユーザ名まで認証することも可能にした．

アンケート返信メール自動集計機能の提案

アンケートを取る状況を想定した場合，1 人から多人数に対して送ったたメールに対する返信メールを，1 つに束ねる機能が存在しない．そこで，メールサーバの自分のドメイン内のユーザへのサービスとして，アンケートメールを自動集計する機能を提案した．アンケートの送信者は拡張ヘッダに返信期限と質問項目数を指定するだけでアンケートメールを送ることができ，返信者はアンケートメールに対して，“返信ボタンを押し”，回答を書き込むだけで返信できるように設計した．

5.1.2　ヘッダへの ID 挿入

この 2 つの高機能化は，ヘッダ内にメール固有の ID を挿入することで実現した．ユーザは拡張ヘッダを用いることとし，MTA でヘッダを解析することによって，この付加機能を利用できるようにした．

ハンドシェークでの送信元認証においては，メールヘッダの一部のハッシュ値を ID とし，データベースに保存しておくことで，受信側の問い合わせに応えるようにした．

アンケート返信メール集計においては，“Reply-To:” フィールドに返信用アドレスと共に ID も埋め込むことにより，受信側が返信先アドレスに返信するだけで，どのアンケートメールに対する返信メールかを特定できるようにした．

5.2　今後の課題

各方式の今後の課題はそれぞれの章で述べたが，結論として今後電子メールをより使いやすく，より便利にし，より多目的に使用できるようにしたいと考えている．

電子メールは，最初の RFC が出てから 20 年以上たったが，Web における様々な機能拡張と比較すると，その機能はほとんど変わっていない．メールこれほどまでに普及した背景としては，シンプルで気軽に使えるといった理由が存在するであろうが，このシンプルさを残しつつ，付加機能としてより高度な機能を実現することは是非必要だと考える．

電子メールの便利な点は，相手と時間的に同期してなくても，コミュニケーションを取ることができることである．電話などと異なり，相手が今何をしているかという状況を気にしなくてよい．送信者はメールを送りたいと思った時に送ればよいし，受信者はそれを見たい時に見ることができる．

しかしその反面，スケジュールの調整などの状況を考えると，メンバ全員がある時間に一堂に会すれば素早く決めることが出来るが，電子メールを用いると，時間的に離れているため，人数が増えた場合に大勢の人の中での調整が行いにくい．本システムを応用し，このような多人数での協調作業を支援するシステムについて検討したいと考えている．

謝辞

本研究を進めるにあたって，何度も相談に乗って下さり，数々の有益な助言を頂きました相田仁教授に深く感謝いたします．素晴らしい研究の環境を与えていただき，大変ありがとうございました．

電子メールを，より使いやすく，より便利にするため，研究パートナーとして意欲的に取り組んでくださった川村泰二郎氏に感謝の意を表します．

ここにお世話になった全ての OB の方々のお名前を挙げることはできませんが，その中で特に，猪瀬・斉藤・相田研時代からの良き環境を残してくださいました富山 忠宏先生，そして研究，計算機管理，その他研究室生活全てにおいて，現役中のみならず卒業後も数々のアドバイスをしてくださいました前川 智則氏に心より御礼申し上げます．

同学年として，お互い切磋琢磨し，共に苦労を分かち合った大須賀 徹氏，佐藤 大将氏，岩村尚氏に感謝します．

最後に，研究室の皆様と生活面で支えていただいた家族に感謝します．ありがとうございました．

2005 年 1 月 31 日

参考文献

[1] A. S. タネンバウム 著, 水野 忠則, 相田 仁, 東野 輝夫, 太田 賢, 西垣 正勝 訳: “コンピュータネットワーク第 4 版,” 日経 BP, Dec. 2003.

[2] Spam Statistics, Brightmail.
http://www.brightmail.com/spamstats.html

[3] Deborah Fallows: “How It Is Huring Email and Degrading Life on the Internet,”
http://www.pewinternet.org/

[4] ガートナー ジャパン株式会社: “Gartner News Release,”
http://www.gartner.co.jp/press/pr20030909-01.pdf, Sep. 2003.

[5] J. Klensin: “Simple Mail Transfer Protocol,” IETF RFC2821, Apr. 2001.

[6] N. Freed, Innosoft, et al.: “Multipurpose Internet Mail Extensions MIME) Part One: Format of Internet Message Bodies,” IETF RFC2045, Nov. 1996.

[7] Anti-Phishing Working Group (APWG),
http://www.antiphishing.org/

[8] Gartner Inc.: “Gartner Study Finds Significant Increase in E-Mail Phishing Attacks”
http://www4.gartner.com/press_releases/asset_71087_11.html, May 2004.

[9] Eric Allman: “Features: Spam, Spam, Spam, Spam, Spam, the FTC, and Spam,” ACM Queue vol. 1, No. 6, Sep. 2003.

[10] 安東, 河, 安, 康, 北野: “SPAM メール対策における新方式の提案,” DICOMO 2003, No. 203, Jun. 2003.

[11] Anti-Spam Research Group (ASRG),
http://asrg.sp.am/

[12] The White House,
http://www.whitehouse.gov/

[13] “迷惑メール関係施策”
http://www.soumu.go.jp/index.html

[14] OECD Work on Spam,
http://www.oecd.org/sti/spam

[15] Internet Mail Consortium ietf-mxcomp,
http://www.imc.org/ietf-mxcomp/index.html

[16] J. Myers: “SMTP Service Extension for Authentication,” IETF RFC2554, Mar. 1999.

[17] Mail Abuse Prevention System.
http://mail-abuse.org/

[18] Open Relay DataBase. http://ordb.org/

[19] Anti-Phishing Working Group: “Phishing Attack Trends Report,”
http://www.antiphishing.org/APWG_Phishing_Attack_Report-May2004.pdf, May 2004.

[20] ウィリアム・スターリングス 著: “暗号とネットワークセキュリティ,” ピアソン・エデュケーション, Sep. 2001.

[21] J. Levine, A. DeKok, et al.: “Lightweight MTA Authentication Protocol (LMAP) Discussion and Comparison,” IETF Internet Draft draft-irtf-asrg-lmap-discussion-01.txt, Feb. 2004.

[22] G. Feyck: “Designated Mailers Protocol,” IETF Internet Draft draft-fecyk-dmp-02.txt, May 2004.

[23] R. S. Brand and L. Sherzer: “Designated Relays Inquiry Protocol (DRIP),”
http://www.sherzer.net/draft-brand-drip-02.txt, Oct. 2003,

[24] J. Levine: “A Flexible Method to Validate SMTP Senders in DNS,” IETF Internet Draft draft-levine-fsv-01.txt, Apr. 2004.

[25] M. Stumpf and S. Hoehne: “Marking Mail Transfer Agents in Reverse DNS with TXT RRs,” IETF Internet Draft draft-stumpf-dns-mtamark-01.txt, Feb. 2004.

[26] H. Danisch: “The RMX DNS RR and method for lightweight SMTP sender authorization,”
http://www.danisch.de/work/security/txt/draft-danisch-dns-rr-smtp-03.txt, Oct. 2003.

[27] J. Levine: “Selective Sender,”
http://www.taugh.com/mp/ss.html, Jan. 2004.

[28] M. Lentczner and M. W. Wong: “Sender Policy Framework,”
http://spf.pobox.com/spf-draft-200404.txt, Apr. 2004.

[29] Microsoft Corp.: “Caller ID for E-mail,”
http://www.microsoft.com/mscorp/twc/privacy/ spam_callerid.mspx

[30] P. Resnick: “Internet Message Format,” IETF RFC2822, Apr. 2001.

[31] M. Wong and M. Lentczner: “The SPF Record Format and Sender-ID Protocol,” IETF Internet Draft draft-ietf-marid-protocol-03.txt, Aug. 2004.

[32] Mark Delany: “Domain-based Email Authentication Using Public-Keys Advertised in the DNS (DomainKeys),” IETF Internet Draft draft-delany-domainkeys-base-01.txt, Aug. 2004.

[33] Tumbleweed Communications Corp.: “Using Digital Signatures to Secure Email and Stop Phishing Attacks,”
http://www.tubleweed.com/

[34] Patrick Pantel and Dekang Lin: “SpamCop– A Spam Classification & Organization Program,” Proceedings of AAAI-98 Workshop on Learning for Text Categorization, Mar. 1998.

[35] Mehran Sahami, Susan Dumais, David Heckerman and Eric Horvitz: “A Bayesian Approach to Filtering Junk E-Mail,” Proceedings of AAAI-98 Workshop on Learning for Text Categorization, Mar. 1998.

[36] Bill Yerazunis: “Sparse Binary Polynomial Hash Message Filtering and The CRM114 Discriminator,” Proceedings of 2003 Spam Conference, Jan. 2003.

[37] D. J. Bernstein: qmail
http://cr.yp.to/qmail.html

[38] qmail-smtpd-auth
http://members.elysium.pl/brush/qmail-smtpd-auth/

[39] cmd5checkpw
http://members.elysium.pl/brush/cmd5checkpw/

[40] 石井 裕 著: “CSCW とグループウェア,” オーム社, Apr. 1994.

[41] Mail Dealer
http://www.maildealer.jp/index.html

[42] fml project
http://www.fml.org/

[43] 梅垣 まさひろ, 寺村 綾子 著: “fml メーリングリスト管理,” オーム社, Jun. 2000.

発表文献

[44] 光田 智史, 山本 智幸, 高橋 桂太, 苗村 健, 原島 博: “実時間奥行き推定を用いたインテグラルフォトグラフィからの自由視点画像合成,” 3 次元画像コンファレンス 2003, pp. 13 – 16, Jul. 2003.

[45] Satoshi Mitsuda, Tomoyuki Yamamoto, Keita Takahashi, Takeshi Naemura and Hiroshi Harashima: “Interactive View Synthesis from Integral Photography Using Estimated Depth Information,” Proc. SPIE Three-Dimensional TV, Video, and Display II, Vol. 5243, pp. 116 – 124, Orland, USA. Sep. 2003.

[46] 光田 智史, 相田 仁: “電子メールの送信元認証におけるハンドシェーク活用の提案,” 情報処理学会 GNWS2004, pp. 3 – 8, Jul. 2004.