コンピュータセキュリティ　4－11
（1999．3．5）

電子メールの自動暗号化処理サーバの構築（2）

永江由紀子† 畑中雅彦† 田島和典†† 青木貴††
† 室蘭工業大学
†† ニッテツ北海道制御システム（株）

現在、多数の暗号化電子メール・パッケージが公開・製品化されている。しかし、暗号化鍵の管理等、利用者にとって複雑な操作も多い。そこで、利用者の操作を簡易化することを目的に、暗号化処理サーバの構築を検討した。今回報告する暗号化処理サーバは、利用者の個人名ではなくメール・システムを所有するサイトのメール・ドメイン名に基づいて、自動的に暗号化／復号化処理の可否を判断する。本稿では、メール・ドメイン名に基づいた暗号化／復号化処理の有効性について検討する。また、今回試作を行なった暗号化処理サーバの構成・動作および動作確認実験の結果についても報告する。

A Construction of Automatic Cryptograph Server for Mail (2)

Yukiko NAGAE†, Masahiko HATANAKA†
Yasunori TAJIMA††, Takashi AOKI††
† Muroran Institute of Technology
†† Nittetsu Hokkaido Control System Co., Ltd.

There are many implementations for encrypted mail, but most of them require tedious and trouble-some key management jobs of every end user. For the purpose of easier use, we have studied an automatic cryptograph server for mail. Depending on the domain names rather than on the user names, this server decides whether a given mail is coded (decoded) or not, and it automatically does the mail using the key determined by the domain. In this article, we discuss the several advantages of this type server and show our prototype server implementation.

1 はじめに

インターネットを利用した情報通信の普及にともない、暗号化技術を用いたインターネット上の通信セキュリティ対策が注目されている。特に、電子メール・サービスは個人的な情報を送受信することが多く、盗聴に弱いシステム構成を持つことから、電子メール本文の暗号化処理は有効なセキュリティ対策である。このため、現在、多数の暗号化電子メール・パッケージが公開・製品化されている[1]。しかし、これらの多くは個人を対象とした汎用パッケージであり、暗号化処理の可否の判断や不特定多数の暗号化鍵の収集・管理等には利用者の介在が必要となる。また、利用者がサービスを要求するたびにユーザ認証を求められる場合も多く、このことが利用者の負担となる場合もある。

そこで、複数のユーザが一つの電子メール・サーバを使用して電子メールの送受信を行うLAN（Local Area Network）環境等のメール・システムに着目し、既存のメール・システムに

暗号化処理サーバを追加することによって、暗号化処理に付随する利用者の負担を軽減させることを検討してきた。その一貫として、以前に SMTP （Simple Mail Transfer Protocol）サービスに対する proxy サーバとして動作し、メール・ドメイン名（メール・アドレスの"@"以下の文字列）に基づいて自動的に暗号化処理の可否を判断する暗号化処理サーバを検討・試作した [2]。また、インターネット環境を用いた動作確認実験についても報告を行なった [3]。

しかし、以前試作した暗号化処理サーバでは、メール・システムにおける電子メールの流れが送信側と受信側で非対称となっていた。また、To、Cc、Bcc を用いて暗号化対象と非対象の宛先を併記し、複数の同文電子メールを送信する場合の対応が不十分であった。このため今回、暗号化／復号化の機能を、二つのサーバ・プログラムで別々に処理することで、メール・システムにおける非対称性を解決した。二つのサーバ・プログラムは、それぞれ SMTP／POP（Post Office Protocol）サービスに対する proxy サーバとして動作する。さらに、暗号化対象および非対象の宛先を併記した電子メールを送信する場合の対応も行なった。

本稿では、メール・ドメイン名に基づいて暗号化処理の可否を判断することの利点について考察する。また、今回試作を行なった電子メールの自動暗号化処理サーバの構成・動作および動作確認実験の結果について報告する。

2 メール・ドメイン名に基づく判断

一般的なメール・システムでは、利用者個人の間で電子メールの暗号化／復号化処理を行なう。すなわち、暗号化／復号化鍵は利用者個人が所有する。このため、複数の人に暗号化電子メールを送信する場合には各通信相手ごとに固有の暗号化鍵が必要になり（図 1 参照）、受信する場合も同数の復号化鍵が必要になる。さらに、暗号化／復号化鍵の管理や暗号化処理の可否の判断が利用者個人に委ねられるために、秘密鍵を忘れる、公開鍵の収集が繁雑である等の問題点に加えて、サイト（例えば、企業）の方針によって必要とされる電子メールの暗号化処理が個人の裁量により正常に行なわれない等の問題も生ずる。

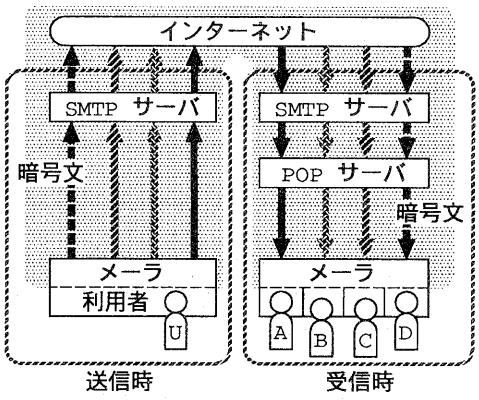

図 1: 一般的なメール・システム

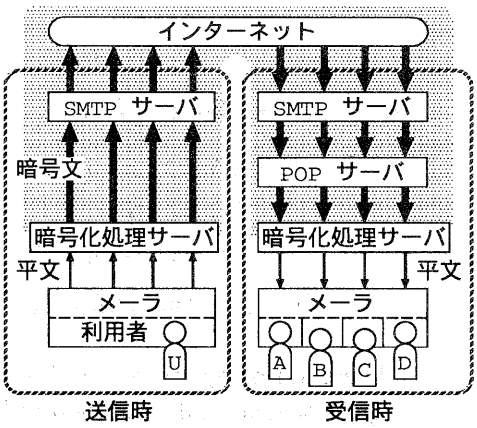

図 2: 目的とするメール・システム

そこで、既存のメール・システムに暗号化処理サーバを追加することによって、インターネット上の特定のサイト間でのみ、電子メールの暗号化を行なうシステムを提案する。このシステムでは、暗号化処理サーバ間で電子メールの暗号化処理を行ない、暗号化鍵は暗号化処理サーバが所有する。また、暗号化／復号化可否の判断は、暗号化処理サーバが電子メールの送信先／送信元のメール・ドメイン名（メール・アドレスの"@"以下の文字列）に基づいて自動的に行なう。

この方法によって、同じ暗号化処理サーバを通して同じメール・システムを利用する複数人

に対して電子メールを送信する場合には、一つの暗号化鍵で電子メールを暗号化処理することが可能となり（図 2 参照）、管理対象となる暗号化鍵の数を減少させることが可能となる。また、利用者個人が電子メールの暗号化処理のために特別な操作を行なう必要はなく、電子メールは暗号化処理サーバで自動的に暗号化／復号化処理される。

暗号化鍵の管理は暗号化処理サーバの管理者に委ねられる。これは、個人で管理するよりも、責任ある担当者が行なうことから、より安全が保たれると考える。

この方法では、図 2 に示すように、暗号化処理サーバとメーラの間では、電子メールは平文（暗号化処理をしていない文）のまま流れる。このため、LAN 内部を流れるデータを盗聴された場合には、電子メールの盗聴もまた防ぐことができない。しかし、サイト外部者による盗聴はファイアウォール等で回避できる問題である。また、サイト内部者による盗聴については、本研究で目的としている暗号化処理サーバの適用範囲ではないと考えている。

3 仕様

実装のため、以下のような仕様を定めた。

1. 暗号化／復号化処理の可否の判断は、送信先／送信元アドレスのメール・ドメイン名に基づいて行なう。
2. 暗号化対象サイトに対するすべての電子メールを暗号化／復号化処理する。
3. 既存の電子メールシステムをできるだけ変更せず、本サーバの単純な追加により全体が機能することを考慮する。
4. To、Cc、Bcc を使用して、暗号化対象および非対象のサイトのメール・ドメイン名を含む宛先を併記する場合を考慮する。
5. サイトごとに、暗号化アルゴリズムを変更できるようにするため、複数の暗号化アルゴリズムに対応する。
6. 暗号化／復号化処理対象になるサイトの情報や暗号化鍵に関する情報のセキュリティを考慮する。
7. モバイル端末における個人用暗号化プログラムとの連携を考慮する。

1、2、3 は、提案する暗号化処理サーバの最大の目的であり、本サーバの特徴となる部分である。また、4、5、6、7 に関しては、実装にあたって考慮した点である。しかし、5, 6, 7 に関しては具体的な仕様を検討中であり、実装は行なっていない。

4 実装

4.1 開発言語と開発環境

本サーバの実装には、Java 言語を使用し、開発環境には JDK（Java Developers Kit）の 1.1.5 を使用した。これは、Java 言語ではネットワーク・プログラミングが容易であり、移植性にも優れていると考えられること、また、暗号化ルーチンを別クラスで定義することによって、暗号化アルゴリズムの変更を容易に行なうことが可能になると考えられるためである。

4.2 暗号化アルゴリズムとライブラリ

電子メール本文の暗号化アルゴリズムとして、対象鍵暗号方式 [4] の暗号化アルゴリズムの一つである IDEA（International Data Encryption Algorithm）[5] を採用した。また、可視コード化には Base64 方式を採用した。可視コード化とは、8 ビット・コードを 7 ビット・コードに変換する処理のことである。暗号化処理後のデータは任意の 8 ビット・コードを含むため可視コード化が必要になる。これらの暗号化クラスおよび可視コード化クラスの実装には、Systemics 社が開発した Cryptix V2.2 暗号アルゴリズムクラスライブラリ*を使用した。

暗号化鍵等の暗号化情報の保管方法については検討中であることから、今回の実装では電子メール本文の暗号化／復号化処理のための暗号化鍵には、プログラム中に組み込んだ文字列を使用した。

4.3 暗号化可否の判断

暗号化可否の判断に用いるメール・ドメイン名は、送信時には SMTP コマンド [6] の引数

*http://www.systemics.com/

から取得し、受信時には、メール・ヘッダに記述された送信者のメール・アドレスから取得する。判断時に参照する暗号化対象サイトの情報は予め平文のファイルとして保存しておき、処理実行時に読み込む。

4.4 暗号化処理サーバの proxy サーバ化

既存の電子メール・システムをできるだけ変更せずに暗号化処理サーバを追加することによって全体が機能することを考慮して、SMTP サーバおよび POP サーバの proxy サーバとして動作するための実装を行なった。暗号化処理サーバを追加する場合に必要な処理としては、メーラに設定された SMTP および POP サーバのホスト名を、暗号化サーバを実行するホスト名に変更することがある。

4.5 宛先の複数併記への対応

To、Cc、Bcc を用いて複数の宛先に同文の電子メールを送信する場合にも、それぞれの宛先ごとに暗号化／復号化処理の可否を判断し、暗号化／復号化処理を行なう。しかし、同文の電子メールを、一方は暗号化し一方は平文で送信することには、セキュリティ上問題があると考えられる。このため、今回は実装していないが、変則的な電子メールが送信された場合に管理者または利用者個人に対して警告を送る等のセキュリティ方針に基づいた処理を行なうことを検討している。

5 構成

5.1 メール・システム全体の構成

メール・システム全体の構成および電子メール送受信時における処理の流れを図 3 に示す。

ただし、実装した暗号化処理サーバを本稿では Crypt - Mail Secretary と呼び、以降の図中では簡易化のため CMS と略称している。

実装した暗号化処理サーバは二つのサーバ・プログラムによって構成されている。メーラと SMTP サーバとの通信を代理中継する間に、電子メール本文の暗号化処理を行なう CMS - SMTProxy と、メーラと POP サーバとの通信を代理中継する間に、電子メール本文の復号化処理を行なう CMS - POProxy である。

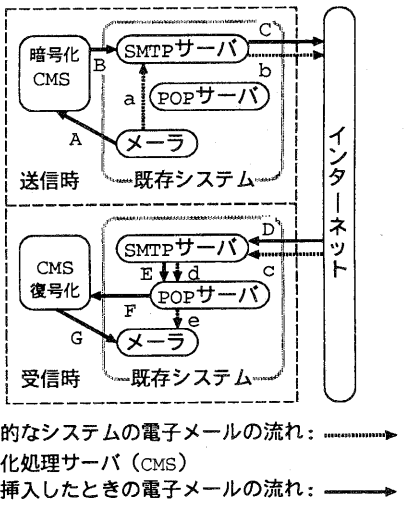

図 3: メール・システムの全体構成図

5.2 CMS - SMTProxy の構成

図 4 に CMS - SMTProxy の構成と処理の流れを示す。

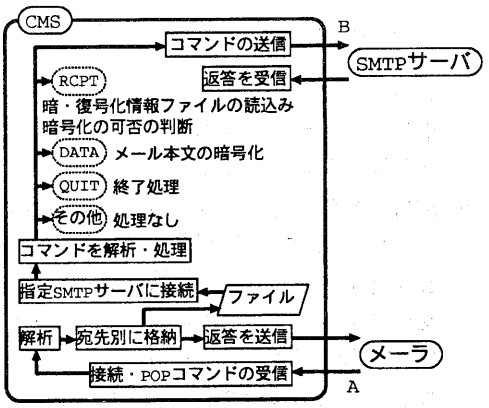

図 4: CMS - SMTProxy の構成図

CMS - SMTProxy ではメーラからの接続および SMTP コマンドを受信すると、SMTP に則った返答を独自に返し、受信したコマンド列を宛先別に格納する。メーラから終了命令を受信すると、CMS - SMTProxy は、指定した SMTP サーバに接続し、格納したコマンド列

を順番に送信する。また同時に、送信先と送信元のメール・ドメイン名を分析し、設定した暗号化情報に従ってメール本文の暗号化処理を行なう。送信を終了した後に、SMTP サーバとの接続を切る。

5.3 CMS - POProxy の構成

図 5 に CMS - POProxy の構成と処理の流れを示す。

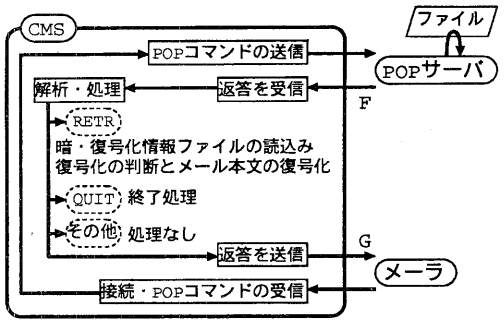

図 5: CMS - POProxy の構成図

CMS - POProxy では、メーラからの接続を受信すると、指定した POP サーバへ接続を行なう。その後、メーラから送られてくる POP コマンド [7] を POP サーバへ中継し、POP サーバから送信される返答をメーラに中継する。特に、POP サーバからメール本文が返答された場合は、送信元のメール・ドメイン名を分析し、設定した暗号化情報に従ってメール本文の復号化処理を行なう。メーラから終了命令を受信すると、終了処理を行ない、POP サーバとの接続を切る。

6 動作確認実験

今回実装した暗号化処理サーバの動作確認を行なうために、電子メールの送受信実験を行なった。

6.1 実験環境

実験を行なうために、学内の LAN 上に三つのメール・システムを用意した（図 6 参照）。

メール・システム A、B は、SMTP サーバ、POP サーバおよび暗号化処理サーバで構成し、メール・システム C は SMTP サーバと POP サーバのみで構成した。これは、暗号化を行なう場合と行なわない場合の双方の動作を確認するためである。本実験環境において各メール・システムの SMTP サーバと POP サーバは同一ホスト上で動作する。

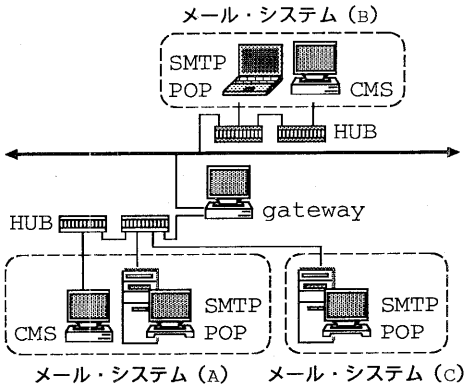

図 6: 実験環境

6.2 使用ソフトウェアと計算機

本実験環境の各メール・システムで使用したソフトウェアと計算機の OS（Operating System）を表 1 に示す。メーラとしては Xemacs-20.3 + im-76 + mew-1.92.4 を使用した。

メール・システム A	OS	ソフトウェア
SMTPサーバ	SunOS4.1.3	Sendmail-8.8.7
POPサーバ		qpopper-2.1.4
CMS	linux2.0.36	JDK-1.1.5

メール・システム B	OS	ソフトウェア
SMTPサーバ	FreeBSD2.2.1	Sendmail-8.8.5
POPサーバ		qpopper-2.53
CMS	FreeBSD2.2.6	JDK-1.1.5

メール・システム C	OS	ソフトウェア
SMTPサーバ	linux2.0.35	Sendmail-8.8.7
POPサーバ		qpopper-2.53

表 1: 使用ソフトウェアと計算機

6.3 実験方法

以下の二種類の方法で実験を行なった。

1. メール・システム B からメール・システム A、C のそれぞれの宛先に対して電子メールを送信する。

2. メール・システム B から、メール・システム A、C の宛先を併記することによって同時に双方に対して電子メールを送信する。

送受信した電子メールの内容は三種類である。すなわち、仮名・漢字を含むメール、英数字のみを含むメール、バイナリ・ファイルを添付したメールである。バイナリ・ファイルとしては GIF（Graphics Interchange Format）の画像ファイル（約 36 KBytes）を使用した。

暗号化／復号化処理が正常に動作したか否かは、暗号化処理サーバがメーラおよび SMTP／POP サーバとの間で送受信したデータを、標準出力に出力することによって確認した。

6.4 実験結果

動作確認の結果、三種類全ての電子メールについて想定した動作結果を得ることができた。すなわち、暗号化／復号化処理対象として設定したメール・システム A に送信した電子メールは正常に暗号化／復号化処理されることが確認できた。また、暗号化／復号化処理非対象として設定したメール・システム C に送信した電子メールは平文のまま転送されることが確認できた。

7 まとめ

今回報告を行なった暗号化処理サーバでは、利用者個人の認証を省略し、メール・ドメイン名に基づいて自動的に暗号化／復号化処理の可否を判断し、処理を行なう。このため、暗号化処理サーバおよびその管理者に対する利用者の信頼が前提条件となるが、多数の電子メールが暗号化されずに平文のまま流通している現状においては、大きな問題はないと考えている。それ以上に、利用者個人による操作を全く必要とせずに、暗号化／復号化処理を行なうことの利点は大きいと考える。また試作実装を行なったサーバに関しては、動作確認実験によって、基本的な機能が正常に動作することを確認することができた。

今後の予定として、インターネット環境を使用した動作確認実験と、動作の安定性を調査するための運用実験を行なうことを検討している。さらに、今回実装を行なわなかった部分、すなわち仕様で列挙した、複数の暗号化アルゴリズムへの対応、暗号化情報のセキュリティ対策、モバイル端末における個人用暗号化プログラムとの連携等について具体的な仕様を検討することを考えている。

現在検討中の案としては以下がある。

- 複数の暗号化アルゴリズムに対応するため、IDEA 以外の暗号化アルゴリズム・ライブラリを用いた暗号化ルーチンの作成と、暗号化情報ファイルに暗号化アルゴリズム指定情報の追加を行なう。
- 暗号化情報を管理するために、Java 言語の hashtable に情報を格納しファイルに保存する。

参考文献

[1] 稲村雄：OPEN DESIGN, 3, 3, CQ 出版社, pp. 62-89 (1996)

[2] 伊藤達也 他："電子メール自動暗号化処理のための簡易サーバの構築", 第3回 インターメディア・シンポジウム Sapporo'98, pp. 40-43 (1998)

[3] 永江由紀子 他："電子メールの自動暗号化処理サーバの構築", 平成10年度電気関係学会北海道支部連合大会講演論文集, p. 421 (1998)

[4] 稲村雄：OPEN DESIGN, 3, 3, CQ 出版社, pp. 16-20 (1996)

[5] 金子敏伸：Interface, 23, 9, CQ 出版社, pp. 116-126 (1997)

[6] Simple Mail Transfer Protocol, Request For Comment (RFC) 821

[7] Post Office Protocol : Version 3, Request For Comment (RFC) 1939