コンピュータセキュリティ　19－1
（２００２．１２．２０）

Crypt-Mail Secretaryへの自動鍵管理機能の実装

太田貴雄† 畑中雅彦†

これまで我々は、個人向け暗号化電子メールパッケージにおける、暗号鍵の管理等の面倒な作業を解消する目的で、ネットワーク組織を単位とする電子メール自動暗号化処理サーバ (CMS: Crypt-Mail Secretary) について、検討・構築・改良を行ってきた。しかし、CMS によるメールの暗号化には、暗号鍵を手動で交換しなければならないため、暗号化対象の組織数が増大するにつれ、鍵管理が煩雑になる問題があった。

そこで、暗号鍵をネットワーク経由で交換し、さらに定期的に暗号鍵を更新する、自動鍵管理機能を CMS に実装した。本稿では、暗号鍵を CMS で自動管理する方法と、自動鍵管理機能の実装について報告する。

Automatic Encryption Key Management function for Crypt-Mail Secretary

Takao OHTA† , Masahiko HATANAKA†

We had been studying automatic cryptograph system for mail to reduce personal troublesome work of mail encryption (ex. management of encryption keys), and constructing some alpha-version mail-proxy systems (Crypt-Mail Secretary: CMS). In a current CMS system, CMS administrator in an encryption-domain must exchange to the encryption keys manually with the ones in other domains. So the number of encryption-domains is increased, the administrator must take a lot of time to exchange encryption keys.

In this report, we propose an automatic encryption key exchange and management procedure for CMS, which is based on IPsec, to reduce labor of CMS administrator. And, we show an implementation of the proposed CMS system which automatically exchanges the encryption key scheduled by each key's date of validity.

1 はじめに

電子メールは、インターネット上において非常によく使われているアプリケーションの一つであるが、盗聴に弱いシステム構成になっている。このため、電子メールを暗号化した上でやりとりすることは、セキュリティを確保する手段として有効である。このことから、電子メールを暗号化するためのパッケージが多数存在する [1]。しかし、これらのパッケージの多くは個人を対象としたものであり、多数の送信相手に対する暗号化の可否判断や暗号鍵の管理などをユーザが行う必要があるため、ユーザにとって負担となっていた。

通常、企業や大学などの組織では、複数のユーザが一つの電子メールサーバを用いて、電子メールの

† 室蘭工業大学　Muroran Institute of Technology

やりとりを行っていることが多い。そこで我々は、電子メールに関するこの利用形態に着目し、既存のメールシステムに大きな変更を加えることなく、組織単位での暗号化・復号化を自動的に行うメール・プロキシ・サーバ (Crypt-Mail Secretary: CMS) について検討・試作 [2] および改良を行ってきた [3][4]。

CMS を利用する利点として、

- 組織で定めるポリシーに基づき、組織間を流れるメール暗号化を徹底出来る。
- 対象をユーザから組織にすることで、管理すべき暗号鍵の数を減らすことが出来る。
- ユーザはメールの暗号化・復号化を意識する必要がない。
- ユーザが個別にメール暗号化パッケージを利用する際の妨げにならない。

といったことが挙げられる。

CMS には、暗号化アルゴリズムや暗号鍵についての情報を保持・管理する機能も含まれているが、CMS を使用して暗号化メールをやりとりする組織間で、これらの情報を事前に手動で相互に交換する必要がある。このため、メールをやりとりする組織の増加に伴い、鍵情報の交換・管理に要する労力が増大する。また、そのため長期にわたり鍵情報が更新されずに、セキュリティが低下する恐れもある。

そこで、CMS に自動鍵管理機能を持たせるために、既に自動鍵交換機能を実現しているセキュリティアーキテクチャである、IPsec (Internet Protocol Security) に着目した。IPsec は、ネットワークでやりとりするデータを、IP パケット単位で暗号化・認証することでセキュリティを確保する技術である [5]。我々は、IPsec の自動鍵交換機能を用いて、CMS 同士が自動で暗号鍵をネットワーク経由で交換する機能を実装した [6]。

さらに、CMS の暗号情報データベースに暗号鍵の有効期限を導入し、自動鍵交換機能を用いて、暗号鍵を定期的に自動更新する機能を実装した。これにより、CMS の暗号鍵の管理を自動化する、自動鍵管理機能を実現することが出来た。

本稿では、我々が考案した自動管理機能の仕組み・動作および動作確認実験について報告する。

2 CMS の概要

本節では、これまで我々が開発してきた CMS の概要について解説する。

2.1 CMS の特徴

CMS の主な特徴として、以下のような点が挙げられる。

- メールアドレスの@以降の部分 (ドメイン名) によって、暗号化処理の有無・暗号化アルゴリズム・暗号鍵を判断する。
- 暗号鍵は CMS が管理する。
- SMTP/POP サーバの Proxy サーバとして動作することで、既存のメールシステムに変更を加えることなく暗号化・復号化処理を追加出来る。

図 1 に一般的なメールシステムの流れと、CMS を追加したメールシステムの流れを示す。メーラから CMS を経由させてメールを送受信することで、利用者は暗号化・復号化を意識することなく、組織単位で暗号化メールを利用することが出来る [1]。

2.2 暗号鍵の管理

メールの暗号鍵は、メールをやりとりする組織の CMS 管理者同士が手動で交換する。交換した暗号鍵は、CMS 暗号情報管理プログラムを用いて、CMS 暗号情報データベースに登録する。CMS 暗号情報データベースには、暗号化アルゴリズムと暗号鍵が、組織ごとに登録されている。また、データベースの内容は CMS の管理者鍵によって暗号化される。

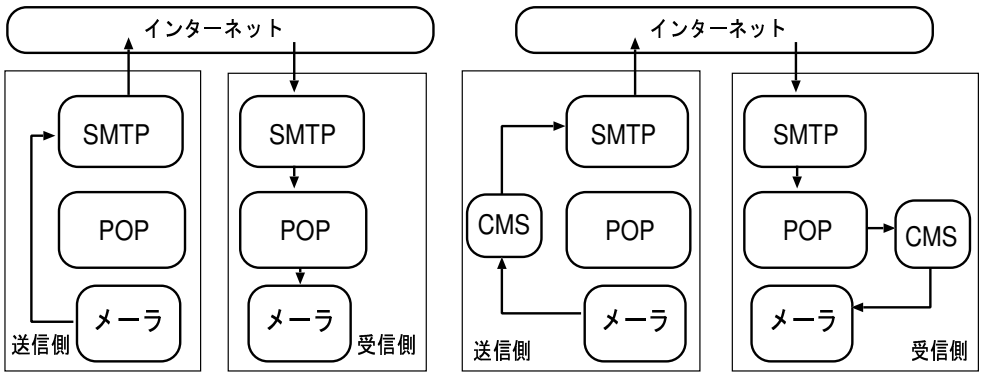

図 1: 一般的なメールシステム（左）と CMS を追加したメールシステム（右）

3　自動鍵管理機能

CMS の自動鍵管理機能は、以下の 2 機能からなる。

- 暗号鍵をネットワーク経由で自動的に交換する機能 (自動鍵交換機能)
- 自動鍵交換機能を利用して、暗号鍵を有効期限に基づいて定期的に交換する機能 (自動鍵更新機能)

以下、この 2 機能について解説する。

3.1　自動鍵交換機能

CMS の自動鍵交換機能は、IPsec の自動鍵交換機能で交換された暗号鍵を、CMS の暗号鍵として流用する形で実装している [5]。

自動鍵交換機能の処理の流れを図 2 に示す。まず、IPsec による自動鍵交換を使用した暗号化通信を行うよう、通信先などの設定を手動で行う。その後、CMS の自動鍵交換ライブラリによって、IPsec の暗号化対象となる通信が行われる。このとき、IPsec の自動鍵交換デーモンが鍵交換を行う。交換された暗号鍵は、SAD(Security Association Database) と呼ばれるデータベースに登録される。

その後、CMS の自動鍵交換ライブラリが SAD から交換された暗号鍵を取り出し、暗号情報管理プログラムを通して、CMS のデータベースに登録する。

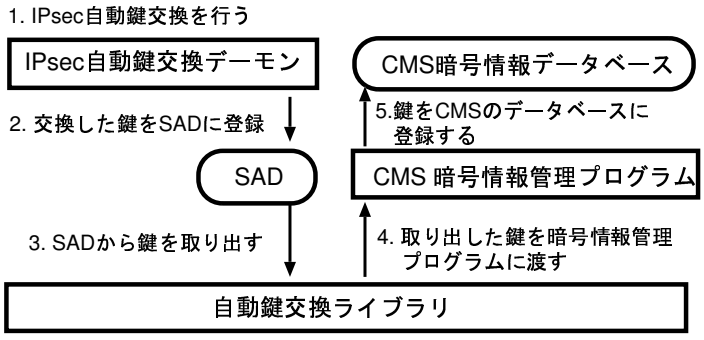

図 2: 自動鍵交換機能の処理の流れ

3.2 自動鍵更新機能

自動鍵交換機能を用いて、CMS の暗号鍵を定期的に更新させる場合、メールサーバに暗号化メールがスプールされた状態で CMS の暗号鍵が更新されると、スプールされた暗号化メールが復号化出来ない問題がある (図 3)。

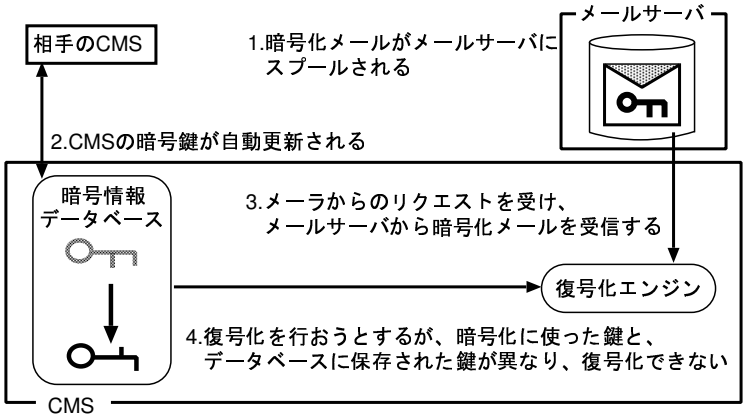

図 3: 自動鍵更新機能でメールが復号化出来ない場合

この問題点を解決するには、以下のような方法が考えられる。

- 方法 1 : 鍵を更新する前に、CMS がメールサーバからメールを受信し、暗号化メールは復号化して CMS 自身にスプールさせた上で鍵の更新を行う (図 4)。
- 方法 2 : CMS が過去の暗号鍵を保持し、復号化するときは保持している鍵の中から使用した鍵を選び出して復号化する (図 5)。

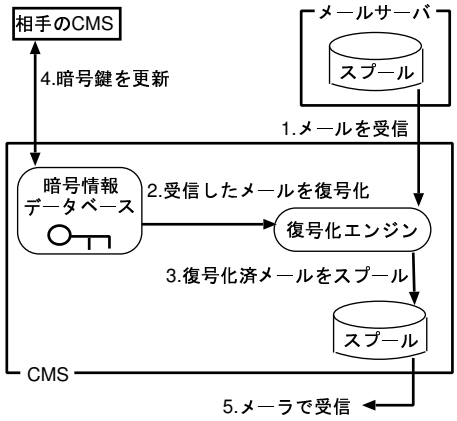

図 4: 方法 1 の場合の処理の流れ

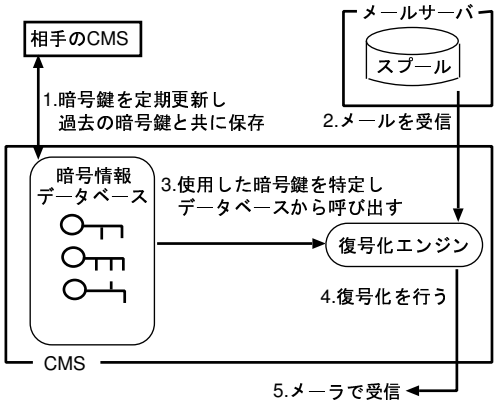

図 5: 方法 2 の場合の処理の流れ

方法 1 では、CMS に全ユーザの POP サーバのパスワードを登録する必要がある。この場合、ユーザがパスワードを変更するたびに、CMS にもパスワードを登録しなければならない。また、メールサーバ

から全てのメールを取り出して CMS にスプールし直すのに、メールのデータ量に応じた時間とネットワークトラフィックが発生する。

方法 2 では、CMS が過去の暗号鍵を保持するため、セキュリティ上望ましくないという問題点がある。

2 つの方法を検討した結果、方法 2 を採用した。過去の暗号鍵を保持することでのセキュリティ上の問題は、CMS の管理鍵を用いて暗号化して保持することで対処する。

4 実装

CMS の開発言語には、ネットワークプログラミングに適した Java を使用している [7]。これは、Java がオブジェクト指向プログラミング言語であるため、暗号化ルーチンとサーバプログラムを別クラスとすることで、暗号化アルゴリズムの変更を容易にすることを目的としている。

一方、自動鍵交換ライブラリは、IPsec のユーティリティプログラムを元に、C 言語で開発されている。IPsec は IP パケット単位での暗号化・復号化を行うため、OS のカーネルなどに依存した実装になっている。このため、IPsec の暗号鍵が登録されている SAD の内容を、Java で作成したプログラムから読み出すことが困難である。そこで、Java から他言語のプログラムを呼び出す JNI (Java Native Interface) を用いて、自動鍵交換ライブラリと CMS 同士が鍵情報をやりとりする。

また、自動鍵交換ライブラリは、IPsec 環境が標準で用意されている FreeBSD 上で開発されており、他の OS での動作は確認していない。

5 動作確認実験

今回実装した自動鍵管理機能の動作確認を行うため、電子メールの送受信実験を行った。

5.1 実験環境

100Base-T で接続された LAN 上に、2 つのメールシステムを用意した (図 6)。なお、本実験環境に置いて、SMTP サーバと POP サーバは同一の計算機で動作する。

使用したソフトウェアを図 7 に示す。メーラには、複数のメールアカウントを扱える Mozilla 0.9.8+ を使用し、1 台の計算機で両方のメールシステムを扱えるようにした。

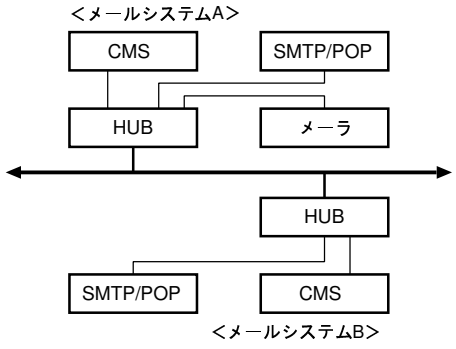

図 6: 実験環境

メールシステム A	OS	ソフトウェア
SMTP/POP	Vine Linux 2.5	Postfix 20010228
		qpopper 3.0.2
CMS	FreeBSD 4.4	JDK 1.1.8

メールシステム B	OS	ソフトウェア
SMTP/POP	FreeBSD 4.6	Postfix 1.1.11
		qpopper 3.1.2
CMS	FreeBSD 4.6	JDK 1.1.8

図 7: 使用したソフトウェア

5.2 実験結果

それぞれの CMS ホストにおいて、IPsec で自動鍵交換を使用した暗号化通信を行うよう設定した。さらに CMS の暗号情報管理プログラムで、相手ドメイン名・暗号化アルゴリズム・鍵の有効期限を設定した。そして、メーラを使い各メールシステム同士でメールの送受信を行った。その結果、暗号鍵が自動的に交換され、メールが正しく暗号化・復号化されて送受信出来ることを確認した。

また、有効期限が切れた状態でメールを送信すると、CMS ホスト間で IPsec の暗号鍵が更新された上で、CMS で使用する暗号鍵が更新された。その後、メールが暗号化されて送信され、正常に復号化して受信出来ることを確認した。

6 まとめ

今回、自動鍵管理機能を実装した CMS は、動作確認実験によって想定した通りの動作結果が得られた。すなわち、IPsec の自動鍵交換機能で交換された暗号鍵が CMS に取り込まれ、正常に暗号化・復号化できること、および有効期限が切れた暗号鍵が、自動的に更新されることを確認した。これにより、従来の CMS で問題になっていた暗号鍵の交換にかかる手間が軽減されると考える。

今後の予定として、現在普及が進んでいる IPv6 に対応させることを検討している。CMS 本体は IPv6 で動作することが確認出来ている [8] が、自動鍵交換プログラムは変更が必要になると考えられる。

また Linux などの他の OS への対応を検討している。今回の実装では、自動鍵交換ライブラリが C 言語で開発されていること、また FreeBSD の IPsec 実装に依存していることから、他の OS では動作せず、Java で開発された CMS の可搬性を損なう原因となっている。他の OS において自動鍵管理機能つき CMS を実現させるために、以下のような方法を検討している。

- OS による IPsec の実装に依存せずに、IPsec 暗号鍵を取り出すプログラムを作成する。具体的には、IPsec の暗号鍵を設定するユーティリティプログラムの出力などから、IPsec の暗号鍵を取得する。
- 暗号鍵をネットワーク経由で平文のまま交換するデーモンプログラムを Java 言語で作成する。その上で、デーモンプログラム間の通信を IPsec により暗号化することで安全に暗号鍵を交換する。

参考文献

[1] Simson Garfinkel 著, 山本和彦 監訳, 株式会社ユニテック 訳 : PGP 暗号化メールと電子署名, 株式会社オーム社, pp. 5-31 (1996)

[2] 永江由紀子 他 : 電子メール自動暗号化処理サーバの構築 (2), 情報処理学会研究報告 99-CSEC-4, pp.61-66(1999)

[3] 永江由紀子 他 : 電子メール自動暗号化処理サーバの構築 (3), 平成 11 年度電気関係学会北海道支部連合大会講演論文集, p.425(1999)

[4] 山口光平 他 : 電子メール自動暗号化サーバの拡張について (4), 情報処理学会研究報告 2001-CSEC-15, pp.1-6(2001)

[5] Naganand Doraswamy, Dan Harkins 著, 林秀幸 監訳, 野村文彦 訳 : IPsec テクニカルガイド, 株式会社ピアソン・エデュケーション, pp. 49-50(2000)

[6] 太田貴雄 他 : 自動鍵交換機能つき Crypt-Mail Secretary の開発, 平成 13 年度電気関係学会北海道支部連合大会講演論文集, p.323(2001)

[7] 有賀妙子, 竹岡尚三 著 : Java 1.1 プログラミング, ソフトバンク社, p.310(1997)

[8] 澤田周 他 : IPv6 対応 Crypt-Mail Secretary の開発, 平成 14 年度開発技術研究会講演要旨集, 発表予定 (2002)