オープンネットワークにおける安全な暗号方式の更新に関する考察

山田竜也[†,☆] 宮地充子[†] 双紙正和[†]

現在，暗号方式は様々なシステムで使われるようになってきた．しかしながら，使用している暗号方式の変更を考慮して構築されたシステムはほとんどない．しかし，暗号方式が攻撃された場合には，攻撃された方式を安全なものに交換する必要がある．一方，暗号方式の更新時には，システム全体の安全性を低下させることなく暗号方式を更新するような新しいスキームを構築する必要がある．本論文では，オープンなネットワークを対象として暗号方式の安全な更新について検討する．まず暗号方式の更新時における致命的な攻撃を述べ，そしてその攻撃に対抗でき，かつ暗号方式を柔軟に変更できるプロトコルを提案する．また，本方式では暗号方式の利用者が個別に更新することとしているが，この場合は各利用者間における暗号方式の同期を維持することが重要となる．そのために，暗号方式の同期をとるためのアルゴリズムについても検討する．

The Secure Renewal of Cryptosystems in the Open Network Architecture

Tatsuya Yamada,[†,☆] Atsuko Miyaji[†] and Masakazu Soshi[†]

Cryptosystems have been really used in various systems, all of which would not be constructed in such a way that a used cryptosystem would be changed to another one. However, if a cryptosystem is attacked, then it must be actually replaced to another cryptosystem. Furthermore, it is important to renew cryptosystems, maintaining the entire system security. Therefore, it is necessary to construct a new scheme in such a way that a cryptosystem is renewed easily without reducing security. In this paper, we investigate a renewal of cryptosystems in the open network architecture. First, we discuss a serious attack in renewing a cryptosystem, then we propose a new protocol system structure that is strong against the attack and enables to replace a cryptosystem flexibly. Our scheme assumes that each entity renews cryptosystems individually. In this case, it is important to keep synchronization of cryptosystems used among entities. Therefore we investigate an algorithm to synchronize them among entities.

1. はじめに

インターネットのようなネットワークにおいては，ElGamal[1]，RSA[8]などのような公開鍵暗号方式を用いて，安全な/認証された通信が行われる．近年，いくつかの暗号方式の弱点が指摘されている：たとえば，鍵長が512ビットのRSA暗号方式が解読されたり[9]，PGP[7]の実装の弱点が指摘されたり[5]，特殊な楕円曲線暗号[6),11]が解読されたりしている．

攻撃された暗号方式は，別の安全な方式に変更する必要がある．つまり，使用している暗号方式を柔軟に変更できるようなシステム構成が必要である．

しかしながら，現在のシステムは暗号方式の更新を想定して構成されていない．このため，暗号方式の更新が必要となった場合，新しい暗号方式を含むシステム全体の再構築が余儀なくされる．これには多大な時間と費用が要求される．これまでの研究では，企業内のネットワークにおけるリニューアル暗号システムが提案されていた[2]．これは，認証局と各エンティティが秘密の共通鍵を共有し，その鍵を利用して暗号方式の更新を行うものである．更新情報の正当性は共有する共通鍵の秘密性に依存する．このため，共通鍵は耐タンパーデバイスに格納することが必須となる．しかしながら，全エンティティが共通の鍵を持ち，それを

† 北陸先端科学技術大学院大学情報科学研究科
Faculty of Information Science, Japan Advanced Institute of Science and Technology
☆ 現在，株式会社東芝SI技術開発センター
Presently with System Integration Technology Center, Toshiba Corporation

耐タンパーデバイスに格納するという仮定はオープンネットワークには適さない．

本論文では，オープンネットワークにおける暗号方式の更新について検討する．システムモデルとして，3つのエンティティ，すなわち暗号方式の正当性を保証する認証局（CA），インターネットへの接続回線を提供するインターネット・サービス・プロバイダ（ISP），そしてISPを通じてインターネットに接続するユーザによる構成を検討する．

このモデルにおいて，各エンティティ間で秘密鍵を共有するという仮定なしに，ネットワーク経由で認証局が暗号方式の更新を通知し，新しい暗号方式のモジュールを配布することを考える．本論文では，このモデルを実現する原始的な更新プロトコルへの攻撃を明確にし，この攻撃を回避しつつ効率的な暗号方式の更新方法を提案する．

ネットワークを経由した暗号方式の配布において問題となる点は，配布者の信頼性と配布された暗号方式（プログラム）の正当性である．つまり，解読された暗号方式の更新情報が認証局から配布される際，この解読された方式を利用して偽の方式を流すという中間侵入者攻撃が可能になるからである．

この攻撃を防ぐために，安全な暗号方式と破られた暗号方式を用いる更新方式を提案する．これによって，破られた暗号方式が利用されたとしても，安全な暗号方式で攻撃を防ぐことができる．さらに，本更新方式は，中間侵入者攻撃を防ぐための最小構成であることを示す．

本論文は次のように構成されている．2章では，ネットワークモデルを定義し，原始的な更新方法とそれに対して考えうる攻撃法を示す．3章では，その攻撃を防ぎ，かつ効率的な新しい更新方法を提案する．4章では，提案方式の安全性について検討する．5章では，提案方式の利点と更新時にかかるコストについて検討する．6章では，各エンティティが個別に更新を行った場合，今まで同期のとれていた暗号方式をいかに同じものに換えるようにするかについて検討する．7章で結論を述べる．

2. システム・モデル

2.1 一般的な暗号システムのモデル

ここでは，暗号方式の更新を可能にするための単純なシステム・モデルを検討する．システム・モデルは3つのエンティティから成る：認証局（CA），プロバイダ（ISP），ユーザである．認証局は暗号方式の完全性を認証し，プロバイダはインターネットの回線を提供し，ユーザはプロバイダに接続するユーザである（**図1**）．これはインターネットにおいて一般的なモデルである．

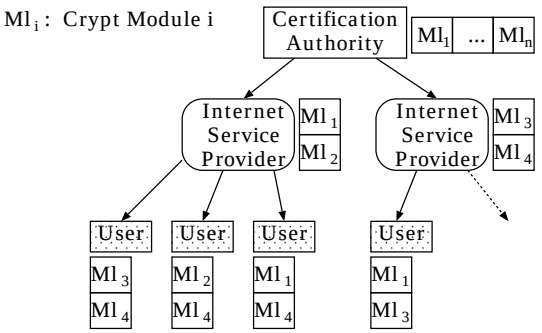

図1 ネットワーク・モデル
Fig. 1 Network model.

このようなモデルにおいては，公開鍵暗号方式と秘密鍵暗号方式によるハイブリッド型の暗号方式がよく用いられる：各ユーザは公開鍵暗号方式の公開鍵と秘密鍵を所持している．データの完全性の確認や鍵共有は公開鍵暗号方式によって行われ，データの暗号化は鍵共有によって交換された鍵を用いた共通鍵暗号方式によって行われる．実際，これはPGPやSSL[4]などによって一般的に行われている．

2.2 暗号モジュールを更新するために必要な条件

Ml_i $(1 \leq i \leq n)$ を認証局によって認証された暗号モジュールとする．各 Ml_i は暗号方式と署名方式の両方を持つものとし，署名生成方式とその検証方式をそれぞれ Sig_i と Ver_i と表す．たとえば，暗号方式と署名方式の対応は次のようになる：RSA暗号方式はRSA署名方式に対応し，有限体 $\mathbf{F}_q$ 上のElGamal暗号方式は同じ有限体 $\mathbf{F}_q$ 上のDSA署名方式[3]に対応する．

認証局は各暗号モジュール Ml_i に対応する公開鍵 $p_{CA}^{Ml_i}$ と秘密鍵 $s_{CA}^{Ml_i}$ を生成し，公開鍵 $p_{CA}^{Ml_i}$ を公開する．各暗号モジュール Ml_i の識別子を Al_i $(1 \leq i \leq n)$ で表し，一方向性ハッシュ関数を H で表す．各暗号モジュール Ml_i について，認証局は署名 $Sig_j(H(Ml_i), s_{CA}^{Ml_j})$ $(\forall j \neq i)$ を生成し，それとともに Ml_i を公開する．認証局の秘密鍵を除いて，すべての変数はシステム内で公開される．認証局は暗号方式の解読情報を入手すると，その情報に署名を付加して各プロバイダに知らせる．一方，プロバイダは暗号モジュールのリニューアルに対して特別の権威を持たないとする．すなわち，認証局から配送されてくる更新情報をユーザに転送する機能しか果たさない．

プロバイダとユーザは少なくとも2個の暗号方式

とそれぞれの暗号方式について認証局の公開鍵を保持している必要がある．また，それぞれの暗号方式は違った系統の暗号方式でなければならない．たとえば，RSA 暗号と ElGamal 暗号や異なる有限体上の楕円曲線暗号というような組合せが望ましい．そして認証局から更新情報を受け取ったとき，新しい暗号方式を取得して更新を行う．

2.3 深刻な攻撃

暗号モジュールの更新時には,「間違った情報」が一番問題になる.「あなたの使用している暗号方式を更新しなさい」という情報は攻撃者につけいる隙を与えることになる．

2.4 原始的な更新プロトコル

原始的な更新プロトコルを示す．暗号モジュールの更新を可能にするために，2.2 節で述べたように各エンティティは少なくとも 2 つ以上の暗号モジュールを所持していなければならない．ここで，暗号モジュール Ml_i $(1 \leq i \leq n)$ があるとし，プロバイダはそのうちの Ml_1，Ml_2 を使用しているとする．そして，Ml_1 が破られたとし，それ以外の $Ml_2, \ldots, Ml_n$ は安全であると仮定する．

認証局とプロバイダ間の原始的な更新プロトコルは以下のようになる．

1. **更新メッセージの配布：** 認証局は暗号モジュール Ml_1 が破れたという情報を入手する．そして，認証局は「暗号モジュール Ml_1 が破られたので Ml_1 以外の別の暗号方式に更新しなさい」というメッセージ M を作成し，それに対して，別の安全な暗号モジュール Ml_j $(j \neq 1)$ による署名の組 $\{Sig_j(H(M), s_{CA}^{Ml_j})\}_{j \neq 1}$ を生成する．次に各プロバイダに署名の組 $\{Sig_j(H(M), s_{CA}^{Ml_j})\}_{j \neq 1}$ とメッセージ M を配布する．
2. **暗号方式の更新：** プロバイダは認証局から送られた署名の組から，自分の使用している暗号モジュール Ml_2 のものを抽出し，検証する．もし，検証が成功したら，暗号モジュール Ml_1 を破棄し，暗号モジュール Ml_2 による署名の付いた別の新しい暗号モジュール Ml_3 をダウンロードしてくる．そして，その署名の検証が成功したら，プロバイダは暗号モジュール Ml_3 をインストールする．

プロバイダとユーザ間の更新プロトコルは以下のようになる．

1. **署名の抽出：** 仮定より，プロバイダはユーザの使用している暗号モジュールを知っている．よって，ユーザの保持する暗号モジュールに対応した認証局の署名を抽出し，それを各ユーザに送付する．
2. **暗号方式の更新：** ユーザは受け取った署名をプロバイダが行ったのと同様に検証し，署名の正当性が確認されたら，新しい暗号モジュールに更新する．

3. 提案プロトコル

本章では我々の提案するプロトコルを記述する．このプロトコルは 2 段階に分けられる：認証局からプロバイダへの配布とプロバイダからユーザへの配布である．

プロバイダの使用している暗号モジュールを Ml_i, Ml_k とし，そのプロバイダに属するユーザの使用している暗号モジュールを Ml_i，Ml_j とする．ここで，Ml_i が破られたとし，Ml_k，Ml_j は安全であると仮定する．議論を簡単にするために，ユーザはプロバイダと同じ暗号モジュール Ml_i を所持しているものとする．しかしもちろん，一般的にはユーザはプロバイダと同じ暗号モジュールを所持している必要はない．

3.1 認証局とプロバイダ間の更新プロトコル

図 2 を参照.

1. **更新メッセージの配送：** 認証局は暗号モジュール Ml_i を更新する必要があるという情報を入手する．そして，認証局は「暗号モジュール Ml_i が破られた」というメッセージ M を作成する．それに対して Sig_i で署名 $Sig_i(H(M), s_{CA}^{Ml_i})$ を生成し，$M' = Al_i||Sig_i(H(M), s_{CA}^{Ml_i})||M$ を作成する．ここで，$Sig_i(H(M), s_{CA}^{Ml_i})$ は，署名方式 Sig_i において秘密鍵 $s_{CA}^{Ml_i}$ による署名を表す．認証局はメッセージ M' に対し，各署名方式 Ml_l $(1 \leq \forall l \leq n,\ l \neq i)$ による署名 $Sig_l(H(M'), s_{CA}^{Ml_l})$ を生成し，$S_l = Al_l||Sig_l(H(M'), s_{CA}^{Ml_l})||M'$ $(1 \leq l \leq n, l \neq i)$ を作成する．そして，認証局はプロバイダにすべての S_l $(1 \leq l \leq n, l \neq i)$ を送る．
2. **配送された署名の検証：** プロバイダが S_l $(1 \leq l \leq n, l \neq i)$ を受け取った後，自分の使用している暗号方式に対応する署名（ここでは，S_k）を取り出す．それを，Al_k，$Sig_k(H(M'), s_{CA}^{Ml_k})$，$M'$ に分割し，認証局の公開鍵 $p_{CA}^{Ml_k}$ によって認証局の署名 $Sig_k(H(M'), s_{CA}^{Ml_k})$ の正当性を確認する．次に，プロバイダは M' を Al_i，$Sig_i(H(M), s_{CA}^{Ml_i})$，$M$ に分割し，認証局の公開鍵 $p_{CA}^{Ml_i}$ によって署名 $Sig_i(H(M), s_{CA}^{Ml_i})$ を検証する．すべての検証が成功したら，プロバイダは「暗号方式 Ml_i が破られた」というメッセージ M の正当性を確認

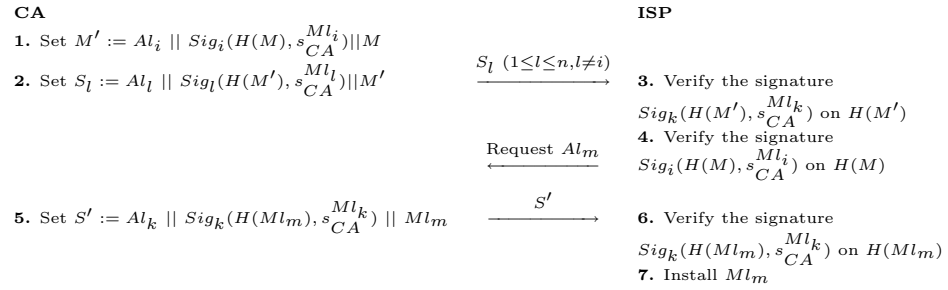

図 2 認証局とプロバイダ間の更新プロトコル
Fig. 2 The renewal protocol between CA and ISP.

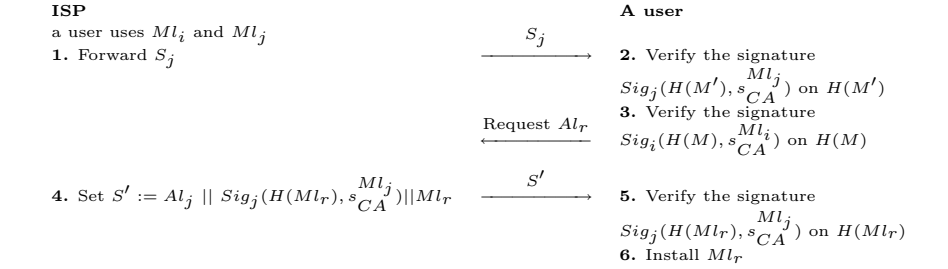

図 3 プロバイダとユーザ間の更新プロトコル
Fig. 3 Renewal protocol between ISP and the user.

する．

3. 暗号方式の更新： プロバイダは Ml_i を破棄し，匿名 FTP サイトなどから自身の持っている暗号モジュール Ml_k による署名のついた別の新しい暗号モジュールを取得する（たとえば，新しい暗号モジュール Ml_m と，それに対する署名 $Sig_k(H(Ml_m), s_{CA}^{Ml_k})$）．そして，その署名 $Sig_k(H(Ml_m), s_{CA}^{Ml_k})$ が正しく検証されたら，Ml_m をインストールする．

注：ここではプロバイダの使用している Ml_i が破られて，Ml_j，Ml_k や他の暗号方式は安全であると仮定した．仮にプロバイダが破られていない暗号モジュール Ml_j，Ml_k を使用している場合，Ml_i を所持していないので利用できる署名 S_l $(1 \le l \le n, l \ne i)$ を見付けることはできない．したがって，プロバイダは暗号モジュールを更新する必要がないと理解する．そして，プロバイダは自分に属しているユーザに対して認証局の署名を転送する役割のみを果たす．

3.2 プロバイダとユーザ間の更新プロトコル

図 3 を参照．

(1) プロバイダはユーザが利用している暗号モジュールを知っていると仮定する．ユーザが暗号モジュール Ml_i，Ml_j を利用していると仮定すると，プロバイダは認証局の署名の中から，S_j をユーザに送る．ただし，ここではプロバイダは認証局の署名を転送するだけの役割しか果たさない．

(2) ユーザは 3.1 節と同様な方法で，認証局の公開鍵でプロバイダから送られてきた署名を検証する．検証をパスすると，ユーザは「Ml_i が破られた」というメッセージ M の正当性を確認できる．

(3) ユーザは Ml_i 以外の新しい暗号モジュール Ml_r を匿名 FTP サイトなどから取得する．そして，3.1 節と同様に Ml_r に添付される署名を Ml_j で検証する．この検証が成功したら，Ml_r をインストールする．

4. 安全性に関する考察

本章では，更新情報の正当性に対して問題となる攻撃を考察し，本提案方式がこの攻撃に対して強いことを示す．

4.1 更新情報の正当性への攻撃

従来方式では，認証局と各エンティティ間で共有する秘密の共通鍵を利用し，暗号モジュールを更新している．このため，各エンティティの共通の秘密鍵の耐タンパー性の仮定は必須であった．今回は，このような共通の秘密鍵を利用せずに暗号方式を更新する．つ

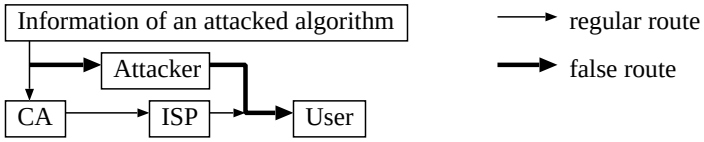

図 4 各エンティティの位置関係
Fig. 4 The position of entities.

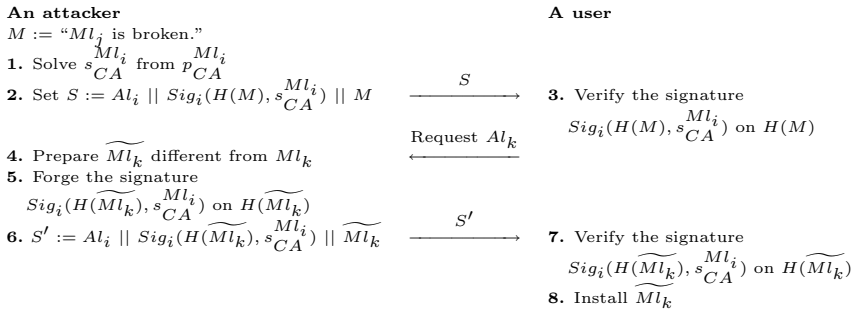

図 5 中間侵入者攻撃
Fig. 5 Intruder-in-the-middle attack.

まり従来の方式における認証局を含む全エンティティの耐タンパー性の仮定を大幅にゆるめ，更新情報を認証局の署名により行い，認証局以外の全エンティティの秘密情報は更新時に利用しない．さらに，認証局の持ついくつかの暗号方式のうち安全な方式を利用する．

本提案方式では，暗号モジュールの更新において，認証局以外の各エンティティ固有の秘密情報は利用しない．そして暗号モジュールの更新時に必要となる情報は認証局の公開鍵だけである．これは公開情報であるため，各エンティティは厳重に管理する必要はない．

公開鍵暗号を用いた暗号モジュールの更新の安全性とは，更新情報の正当性を保証することを意味する．その正当性の保証は認証局の持つ安全な署名方式によってなされる．安全な方式における認証局の秘密鍵が漏洩しないということは，暗号を認証に使用するうえで必要な最低限の条件である．暗号方式の更新情報自体や更新情報に対する署名は公開情報であり，第三者による盗聴などは問題にならない．以上から，更新情報の正当性に対する攻撃は，「認証局へのなりすまし」と「更新情報の改竄」に限られる．それこそが以降で述べる中間侵入者攻撃にほかならない．

4.2 中間侵入者攻撃

認証局は更新する暗号モジュールの情報を各エンティティに配布する．認証局がその情報を配布する前に，攻撃者は間違った情報を配布することにより，攻撃を試みる（**図 4**）．

原始的な更新プロトコルに対する攻撃は三段階で構成される（**図 5**）：

1. **更新情報を改竄する：** 攻撃者はプロバイダとユーザの中間に位置すると仮定する．攻撃者は，解読された暗号モジュール Ml_i を利用して，暗号モジュール Ml_i における公開鍵 $p_{CA}^{Ml_i}$ などの公開情報から秘密鍵 $s_{CA}^{Ml_i}$ を導出する．次に，「暗号方式 Ml_j $(j \neq i)$ が破られた」という偽のメッセージ M に対して，Ml_i を利用して $Sig_i(H(M), s_{CA}^{Ml_i})$ を偽造する．そして，安全な Ml_j を利用しているユーザに $S = Al_i||Sig_i(H(M), s_{CA}^{Ml_i})||M$ を送る．さらに，攻撃者は不正に改竄した暗号モジュール $\widetilde{Ml_k}$ を用意し，暗号モジュール Ml_i による偽の署名 $Sig_i(H(\widetilde{Ml_k}), s_{CA}^{Ml_i})$ を添付して，安全な暗号方式と置き換える．
2. **ユーザが改竄された署名に騙される：** ユーザは S を受け取り，それを Al_i，$Sig_i(H(M), s_{CA}^{Ml_i})$，$M$ に分割する．そして，「暗号方式 Ml_j が破られた」というメッセージ M についての署名 $Sig_i(H(M), s_{CA}^{Ml_i})$ を，Ver_i と認証局の公開鍵 $p_{CA}^{Ml_i}$ で検証する．これによって，ユーザは M を認証局のメッセージだと認識し，安全な暗号方式 Ml_j を破棄してしまう．
3. **ユーザが不正な暗号方式 $\widetilde{Ml_k}$ を取得する：** ユーザは，攻撃者によって改竄された $\widetilde{Ml_k}$ を取得する．そして，その暗号モジュールの署名 $Sig_i(H(\widetilde{Ml_k}), s_{CA}^{Ml_i})$ の検証も成功するので，ユーザは改竄されている不正な暗号モジュールをイン

ストールすることになる．

さらに，上記の攻撃が 2 回以上繰り返されると，正しい暗号方式をまったく持たなくなる．このため，認証局からのメッセージの正当性を検証する手段を失う，という状況になることもありうる．

4.3 攻撃に対する本方式の安全性の評価

Ml_i，Ml_j をユーザの使用している暗号方式とし，Ml_i が破られたと仮定する．したがって，攻撃者は認証局の公開鍵 $p_{CA}^{Ml_i}$ から，秘密鍵 $s_{CA}^{Ml_i}$ を生成することができる．攻撃には以下の 2 通りの方法が考えられる．

(1) 攻撃者は破られた暗号方式 Ml_i を用いてメッセージ M に署名 $Sig_i(H(M), s_{CA}^{Ml_i})$ を偽造する．そして，Ml_j を用いて $Sig_i(H(M), s_{CA}^{Ml_i})$ に対する署名を生成しようとする．しかし，Ml_j は安全な暗号方式なので，Ml_j による署名の偽造は不可能である．

(2) 攻撃者は Ml_j を用いてメッセージ M に署名 $Sig_j(H(M), s_{CA}^{Ml_j})$ をする．そして破られた暗号方式 Ml_i を用いて，$Sig_j(H(M), s_{CA}^{Ml_j})$ に対して署名をする．(1) の場合と同様に，攻撃者は外側の署名 $Sig_i(H(M'), s_{CA}^{Ml_i})$ を偽造することはできるが，Ml_j による内側の署名 $Sig_j(H(M), s_{CA}^{Ml_j})$ は偽造できない．

以上より，いずれの場合も署名の偽造は不可能である．このように，提案プロトコルは中間侵入者攻撃に対して強固であることが分かる．

5. 議　　論

5.1 破られた暗号方式の再利用

システムの信頼性の問題では，あるシステムが故障したときは，そのシステムは使用不可能であるという仮定が一般的である．しかし，本提案方式の場合は，その故障したシステムを暗号方式の更新情報の正当性を保証するのに利用する．もちろん，一般的な意味での故障したシステムというものは利用不可能である．しかし，暗号方式において，“暗号が解読される”，つまり，“暗号が故障する” というのは，暗号方式自体が利用不可能なのではなく，その暗号方式で利用される秘密情報が秘密情報でなくなるということを意味する．したがって，暗号プログラムや，それに付随するデータは依然として，用途により利用可能である．これらの情報を利用し，最小限の資源で暗号方式を更新する．

5.2 提案プロトコルの最小性

本モデルにおいて，各エンティティは少なくとも 2 つの相異なる公開鍵暗号方式を所持していると仮定した．これは，1 つの暗号モジュールが破れたとしても，もう一方の暗号モジュールを利用してモジュールを安全に更新できる最小のモデルである．このモデルにおいて，我々のプロトコルは，安全な更新を実現できる最小の構成となることを示す．

認証局は，Ml_i が破られたとき，その更新のために破られた Ml_i でなく Ml_j と Ml_k という 2 つの安全な暗号方式による署名を生成すると仮定する．この場合，各エンティティが 2 つの暗号方式を持っていて，その 2 つがともに安全であるとすると，暗号モジュールを更新する必要がない．一方，エンティティが安全な暗号方式 Ml_j と破られた暗号方式 Ml_i を持っていた場合，そのエンティティは認証局の署名の検証に必要なもう 1 つの暗号モジュール Ml_k を持っていないので CA の署名を検証することができない．これは，各エンティティは少なくとも 3 個の暗号方式を持っていなければならないのと同義である．このように，本モデルは各エンティティが 2 つの暗号方式を持っているという最小のモデルにおいて安全な更新が可能であることが分かる．

5.3 提案プロトコルの拡張

前章までは，1 つの暗号方式が破れて，もう一方は安全であるという最小のモデルについてのみ議論してきた．本提案方式は，システム内に n 個の暗号方式が存在し，i 個の暗号方式が同時に破れて，それ以外の 1 つが安全であるというモデルにも容易に一般化できる．耐攻撃数を i，すなわち i 個以下の暗号方式が解読されても安全であるようにシステムを設定する．このとき，各エンティティは最低 $i+1$ 個の暗号方式を保持する必要がある．そしてその更新は 3 章と同様の方法で行われる．簡単のため $j\ (\leq i)$ 個の方式，$Ml_1, \cdots, Ml_j$ が解読されたとする．このとき，認証局は破れた j 個の方式でそれぞれ署名をして $Sig_1, \ldots, Sig_j$ を作成し，それに対して安全な暗号モジュール $Sig_k\ (j < k \leq n)$ で二重に署名をして $j(n-j)$ 個の署名

$$\{Sig_k(Sig_l)\}\ (1 \leq l \leq j, j < k \leq n)$$

を作成する．これをプロバイダなどに送付し，それぞれ検証することによって更新が可能となる．

一方，単純に考えられる署名の拡張方法として，破れた方式 $Sig_l\ (1 \leq l \leq j)$ で多重に署名をし，それに対して，安全な方式 $Sig_k\ (j < k \leq n)$ でそれぞれ重ねて署名をする方法が考えられる．つまり，

$$\{Sig_k(Sig_j(\cdots(Sig_1)))\}\ (j < k \leq n)$$

という署名の集合を作成する．しかし，この署名を検証するためには，すべてのユーザがすべての破れた

表 1 コスト
Table 1 Costs.

	認証局	プロバイダ	ユーザ
署名回数	$2j(n-j)$	—	—
検証回数	—	$2j(i+1-j)+j$	$2j(i+1-j)+j$
通信（送信）量	$j(n-j)$	$j(i+1-j)+j$	—
所持する方式の個数	n	$i+1$	$i+1$

j 個の署名方式 $Ml_1, \cdots, Ml_j$ を保持しなければならない．たとえば，ユーザが 1 つの破れた方式と安全な方式を持っている場合，更新できないことを意味する．つまり，単純な拡張では，ユーザの保持する方式が $j+1$ 個では更新できないことを意味し，提案方式に比べて非効率な更新方法である．

5.4 更新にかかるコストの概算

ここでは，暗号モジュールの更新にかかる計算コスト，通信コスト，記憶コストについて，それぞれ検討する．

システム内に存在し，認証局にその正当性を保証されている暗号モジュールの個数を n 個，そのうちで同時に破られる可能性のある暗号モジュールの個数を j 個と仮定したシステムでの更新を想定する．**表 1** は，認証局が署名する回数，プロバイダ・ユーザが検証する回数，送信する署名の個数，各エンティティが所持しなければいけない暗号方式の個数を表す．

現実的には i の値は小さいと考えられ，またその場合 j の値もそれほど大きくなくてよい．よって，更新時には認証局に最も負荷がかかると考えられる．したがって，更新に適した暗号方式とは，署名生成の負荷が軽くできる離散対数ベースのもの[3),10)]が望ましいといえる．

6. 更新時における暗号モジュールの同期の維持

6.1 暗号方式の選択アルゴリズムの必要性

これまでに述べてきた方式によって，各エンティティが独立して暗号モジュールの更新を行うことが可能になった．しかし，更新する以前には同じ暗号モジュールを利用して秘密通信を行っていたエンティティどうしが，更新することによって，共通する暗号モジュールをそれぞれまったく別のものに更新したとする．すると，それまで可能だった公開鍵暗号による秘密通信が更新によって不可能になる．この場合，再び秘密通信を行うためには各エンティティどうしが通信し，使用する暗号モジュールの同期をとる必要がある．しかし，これは更新によって新たな通信コストを発生させる．

一方，各エンティティが使用する暗号モジュールは適度に分散していることが望ましいので，更新によりシステム内で使用する暗号モジュールが偏ることも避ける必要がある．次にその暗号モジュールを更新する必要がでてきたとき，システム全体としての負荷が高くなるからである．

6.2 暗号方式の決定アルゴリズム

以下に暗号方式が破れたときに，認証局が次に推薦する暗号方式を決定するためのアルゴリズムを示す．

準備 認証局は，各暗号方式をネットワークにおける利用者の数が少ないものから順に並べ，利用者の数が等しい方式どうしでは登録した時期の古いものが先頭にくるリストを作成しておく．これは公開情報である．

(1) 破れた方式をそのリストから削除する．

(2) 各エンティティが次に選択する暗号方式は，作成したリストにおいて，利用者の数が最も少く，かつ最も古い時期に登録されたものを選択する．もし，選択したものが自分の利用している暗号方式である場合は，

(a) 利用者数が同じ暗号方式があるならば，次に登録時期が古いものを選択する．

(b) 利用者数が同じ暗号方式がない場合は，次に利用者数が多い暗号方式で最も登録時期が古いものを選択する．

(3) リストをアップデートする．

以上のアルゴリズムによって，開始時に利用者が分散していた場合は，更新後でも利用者が分散するようになる．

7. 結　　論

本論文では，システムの安全性を低下させることなしに，弱い暗号方式を別のものに更新するシステムの構造を示した．また，破られた暗号方式と安全な暗号方式による二重署名の更新プロトコルを提案した．本提案プロトコルは各エンティティに対して最小のコストで破られた暗号方式から安全なものに更新できる．さらに，各エンティティが個別に更新したときに，暗号方式の同期を実現する暗号アルゴリズム決定方式を提案した．

参 考 文 献

1) ElGamal, T.: A public key cryptosystem and a signature scheme based on discrete logarithms, *IEEE Trans. Information Theory*, Vol.31, pp.469–472 (1985).
2) Endoh, N., Okada, K., Okamoto, E. and Tochikubo, K.: Renewable Authentication and Encryption Systems, *CSS'99* (1999).
3) Federal Information Processing Standard (FIPS): Digital Signature Standard, Technical Report, National Institute of Standards and Technology, US Department of Commerce, Washington D.C. (1994). Publication 186.
4) Freier, A.O., Karlton, P. and Kocher, P.C.: *SSL Protocol V.3.0*. http://home.netscape.com/eng/ssl3/ssl-toc.html (1996).
5) Ishizuka, H., Sakai, Y. and Sakurai, K.: On Weak RSA-Keys Produced from Pretty Good Privacy, *Information and Communications Security*, LNCS, Vol.1334, pp.314–324, Springer-Veralg (1997).
6) Menezes, A., Okamoto, T. and Vanstone, S.: Reducing elliptic curve logarithms to logarithms in a finite field, *IEEE Trans. Information Theory*, Vol.39, pp.1639–1646 (1993).
7) Network Associates, Inc.: *Pretty Good Privacy*. http://www.pgp.com/.
8) Rivest, R., Shamir, A. and Adleman, L.: A Method for Obtaining Digital Signatures and Public-Key Cryptosystems, *Comm. ACM*, Vol.21, No.2, pp.120–126 (1978).
9) RSA Laboratories: *Factorization of RSA-155*. http://www.rsa.com/rsalabs/html/rsa155.html (1999).
10) Schnorr, C.P.: Efficient signature generation by smart cards, *Journal of cryptology*. Vol.4, pp.161–174 (1991).
11) Semaev, I.A.: Evaluation of discrete logarithms in a group of p-torsion points of an elliptic curve in characteristic p, *Mathematics of computation*, Vol.67, pp.53–356 (1998).

(平成 11 年 11 月 30 日受付)
(平成 12 年 6 月 1 日採録)

山田 竜也

2000 年 3 月北陸先端科学技術大学院大学情報科学研究科博士前期課程修了．同年 4 月（株）東芝入社．現在，情報セキュリティ技術の研究開発に従事．

宮地 充子（正会員）

1988 年大阪大学理学部数学科卒業．1990 年同大学院修士課程修了．同年，松下電器産業（株）入社．1998 年北陸先端科学技術大学院大学・情報科学研究科助教授．現在に至る．情報セキュリティの研究に従事．博士（理学）．SCIS93 若手論文賞，科学技術庁注目発明賞各受賞．電子情報通信学会会員．

双紙 正和（正会員）

1993 年 3 月東京大学大学院理学系研究科情報科学専攻修了．電気通信大学大学院情報システム学研究科博士後期課程，同研究科助手を経て，1999 年 4 月から現在まで，北陸先端科学技術大学院大学情報科学研究科助手．セキュリティモデル，アクセス制御，分散システムの研究に従事．博士（工学）．