再暗号化依頼可能なDaaSモデルの提案

野口　宏†　小澤　哲†

† 茨城大学　〒316–8511 茨城県日立市中成沢町 4–12–1
E-mail: †{noguchi,ozawa}@mx.ibaraki.ac.jp

あらまし　データベースをアウトソーシングをするためには暗号化が必須であるが，暗号化の鍵を不定期に鍵を変更する必要性が生じる．鍵の変更に伴う再暗号化のコストは多大なものとなるため，暗号化の鍵を知らせることなく再暗号化のサービスを取り入れたDaaSモデルのフレームワークを提案する．

キーワード　DaaS, 再暗号化, revocation

A proposal of re-encryption and revocation method for DaaS

Hiroshi NOGUCHI† and Satoru OZAWA†

† Ibaraki University　Nakanarusawa 4–12–1, Hitachi-shi, Ibaraki, 316–8511 Japan
E-mail: †{noguchi,ozawa}@mx.ibaraki.ac.jp

Abstract　It is indispensable to encryt data in database for realizing Database as a Service(DaaS). And encryption key must be able to updated at the period that the client specified. But, the cost of re-encrytion is very high. Therefore, we propose the framework of DaaS model that is enable re-encryption accompanied refreshing key without revealing any information.

Key words　DaaS, re-encryption, revocation

1. まえがき

近年，データベース (DB) が当たり前の様に利用されているが，データベース管理システム (DBMS) や DB を自組織内に構築し管理，　運用するには多大な労力を必要とする．加えて，インターネットの急速な普及により，DB を利用した Web サービスも広く普及し，それにより DBMS のセキュリティを脅かすことが頻繁に起こっている．こうした背景により，DBMS の機能を提供する Database as a Service(DaaS) が提供されている．しかし，DaaS における DBMS の管理者は，更には DaaS のプラットフォームを提供する OS の管理者は，そこに DB を構築する DaaS の利用者にとっては第三者である．このため，DB 内のデータの不正な閲覧や改竄が起こる可能性を可能な限り減らすことが求められている．

そのために，利用にあたっては暗号を用いることが必須となってきた．encrypt-on-wire 方式では，DB 自体は暗号化せずに DB をアクセスする際の通信経路を暗号化して，セキュリティを確保している．一方，encrypt-on-disk 方式では，通信経路ではなく DB に格納するデータを暗号化することにより，セキュリティを確保している．後者は，SNAD [4], Plutus [3], SiRiUS [2], BA-Rev [6] 等で利用されている．データは利用者が予め暗号化しておきシステムに格納するため，システム管理者であっても閲覧することが出来なくなっている．

しかし，暗号化の鍵が漏れてしまうと，システム管理者の権限を持った第三者によるデータの利用が可能となってしまう．加えて，通信経路を暗号化していなかった場合には，鍵を取得した第三者が通信経路上のデータを復号することが可能となってしまう．

また，暗号化の鍵は使い続けることにより第三者に漏れる可能性が高くなるため，定期的に変更することで より安全に DaaS を利用することが可能となる．一方で，企業等では定期的に行われる人事異動などで利用者が交代する事態が必ず生じるため，事態発生時には対象者のデータベースへのアクセス権の変更が必要となる．鍵を変更しないと，本来鍵を知り得ない立場に変更なった者でも知っている状態が発生してしまう．従って，人事異動等の度に鍵を変更する必要が生じる．また，人事異動は定期的なものばかりでなく突発的なものもあるため，それにも考慮しなければならないのが現状である．

一般的には再暗号化のコストは非常に高いのであるが，データの安全性の観点からアウトソース元で行う必要があった．それは，アウトソース先で再暗号化を行うためには，アウトソース先に新しい鍵を知らせなければないためである．そのため，再暗号化を行うには一度平文に戻した後で，新しい暗号鍵を用いて暗号化しなければならない．Reverse Order Re-Encryption (RORE) [7] や鍵の漏洩に対して安全なデータ保管 [5] 等では，再暗号化をアウトソース元ではなくストレージ側で実行し，更

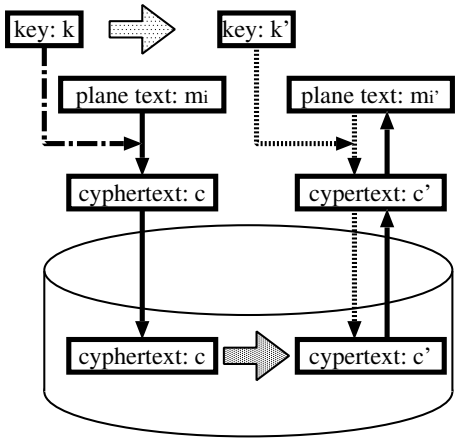

図 1　依頼可能な再暗号化

に平文を生成しないため，安全性を確保し再暗号化をアウトソースすることが可能となった．

そのため本稿では，まず再暗号化の方式の検討を行い，次に暗号化されたデータのアクセスの権限失効の際に行われる再暗号化の手法を検討を行う．その後で，鍵変更時の再暗号化までも安全にアウトソーシング可能となるシステムのフレームワークを提案する．

2. 再 暗 号 化

暗号化を導入した場合，同一の鍵を利用し続けることは非常に危険なことであり，少なくとも一定時間が経過した時に鍵を変更する必要がある．一方で，組織のデータに対して暗号化を導入した場合等は，データの利用に関する権限が変更となることも頻繁に起こる可能性がある．異動は，定常的なものだけではなく突発的なものもあり，定期的に変更するだけでは対応できない．このため，利用者が指定した時に鍵を変更し，利用権限の無いユーザが利用できないようにしなければならない．

また，鍵を変更した場合には，これまでは暗号化されたデータを復号し一時的に平文に戻した後で，新しい鍵で再暗号化をしなければならなかった．アウトソーシング先に暗号の鍵を教えることも，一時的にでも平文に戻る状態になることは，暗号化の導入が無意味なものとなってしまう可能性がある．そのため，図 1 に示すような，暗号文を平文に戻さずに再暗号化を可能とした方式も研究されている．これにより，アウトソーシング先に再暗号化を依頼することが可能となる．

2.1 Forward Secure Encryption

forward　secure encryption [1] では，暗号化のアルゴリズムとして公開鍵暗号方式を前提とし，期間毎に再暗号化をしていくものである．公開鍵 pk は変更せず，秘密鍵を新しくして再暗号化を行う方式である．

しかし，図 2 に示したように，この方式では秘密鍵が漏洩した場合，その後の期間のデータが全て復号できることになってしまう．

また，一定期間毎に再暗号化を行うため，突発的に再暗号化に対応するためには再暗号化のための期間の設定が非常に難しい．

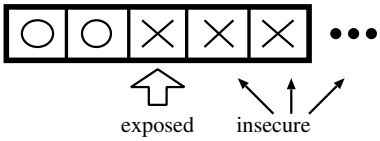

図 2　forward secure encryption の鍵の漏洩

2.2 RORE

RORE では，暗号化のアルゴリズムとして共通鍵暗号方式を前提としているが，加えて暗号化と復号が可逆となることも前提としている．共通鍵暗号の鍵はロックボックス等で管理されることになり，ロックボックスでは公開鍵暗号方式を利用し共通鍵を管理する．

データの暗号化の際に共通鍵暗号方式を利用することにより，利用における計算のコストが公開鍵暗号化方式を利用して暗号化することに比べて大変低い．

一方で，再暗号化をする際に，共通鍵暗号が可逆であることを前提としており，再暗号化の際に新しい鍵で暗号化した後で古い鍵で復号を行っている．このため，鍵を管理するロックボックスの管理者権限を持った者に鍵を知られる可能性がある．このようにして漏れた鍵が第三者に渡った場合，暗号化された暗号文が通信経路上で取得された場合は復号されてしまうことになる．

2.3 鍵の漏洩に対して安全なデータ保管

我々は，鍵の漏洩に対して安全なデータ保管に関するモデルを提案している．暗号化のアルゴリズムは公開鍵暗号方式を前提とし，forward secure encryption とは異なり，データ所有者の意志で再暗号化の時期を決めることができるものである．また，秘密鍵が漏洩したとしても再暗号化した後の暗号文を復号することも，もちろん秘密鍵が漏洩する以前の暗号文を復号することもできないものである．

2.3.1 モ デ ル

データ保管サービスのモデルとしては，サービスプロバイダ $\{P_1, \cdots, P_n\}$ の n 個のサーバで構成されていると仮定する．それぞれの期間 i においてユーザは独自にかつランダムに公開鍵と秘密鍵のペア (pk_i, sk_i) を生成する．この時，古い公開鍵 pk_{i-1} のもとでの平文 m の暗号文 c を，新しい公開鍵 pk_i のもとでの平文 m の暗号文 c' にどの様な情報も漏らすことなく再暗号化させることが可能である．

このモデルでは，公開鍵暗号方式の拡張 ElGamal で実現することが示されているため，以下では拡張 ElGamal 暗号方式を用いて基本的な考えを示す．

2.3.2 拡張 ElGamal 暗号方式

グローバルな情報 I は素数 p, q(但し，$p = 2q + 1$) から成り，位数 q の乗法群 G と G の生成元 g で構成される．

(鍵生成)　鍵生成アルゴリズム $\mathcal{K}(\mathcal{I})$ はランダムに $z \in Z_q$ から選び，$Y = g^z$ を計算する．秘密鍵は $sk = z$, 公開鍵は $pk = Y$ である．

(暗号化)　平文 $m \in G$ が与えられた時，暗号化アルゴリズムは初めにランダムに $r \in Z_q$ を選ぶ．そして，$u = g^r$ と

$e = mY^r$ を計算する．暗号文は (u, e) である．

(復号)　暗号文は (u, e) が与えられた時，復号アルゴリズムは $m = e/u^z$ を出力する．

2.3.3 再暗号化の基本的な考え

この拡張 ElGamal に対して，次の様な再暗号化を考える．

現在の鍵ペアを $(pk, sk) = (Y, z)$ とし，新しい鍵ペアを $(pk', sk') = (Y', z')$ とする．ここで，$Y = g^z$ であり，$Y' = g^{z'}$ である．そして，現在の暗号文の集合を $D = \{(u_1, e_1), \cdots, (u_l, e_l)\}$ とする．ここで，m_i を平文とすると，$(u_i, e_i) = (g^{r_i}, m_i Y^{r_i})$ である．

この時，新しい暗号文の集合を $D' = \{(u'_1, e'_1), \cdots, (u'_l, e'_l)\}$ とすると，新しい暗号文 (u'_i, e'_i) は現在の暗号文 (u_i, e_i) を新しい公開鍵 Y' で再暗号化したものである．ここで $u'_i = u_i$ を仮定する．$e'_i = m_i (Y')^{r_i}$ であるので，e'_i は次の通り e_i より導き出せる．

$$e'_i = m_i (Y')^{r_i} = e_i (Y')^{r_i} Y^{-r_i} = e_i (g^{r_i})^{z'-z} = e_i (u_i)^{z'-z}$$

これにより，基本的には再暗号化のために DaaS に $z' - z$ を与えることにより，再暗号化を依頼することが可能となる．$z' - z$ より $(u_i)^{z'-z}$ が計算可能で，D の各 (u_i, e_i) に対して $e'_i = e_i (u_i)^{z'-z}$ により再暗号化が可能である．この時，敵は $z' - z$ 上の何の情報も得られない。それゆえ，サービスプロバイダ $\{P_1, P_2, \cdots, P_n\}$ のうちどんな $k+1$ 個でも共同で $(u_i)^{z'-z}$ を計算できるように $z' - z$ を $P_1, P_2, \cdots, P_n$ に分配する．

2.3.4 アップデート

$GF(p)$ 上のランダムな多項式 $f(x) = s + a_1 x + \cdots + a_k x^k$ を選ぶ．ここで $s = z' - z \bmod q$ であるとする．$P_j (j = 1, \cdots, n)$ に $f(j)$ を秘密に与える．$y_0 = g^s, y_1 = g^{a_1}, \cdots, y_k = g^{a_k}$ を公開する．各 $(u_i, e_i) \in D$ について以下を行う．

（1） 各 P_j は $v_{i,j} = (u_i)^{f(j)}$ を公開し，零知識証明において $(g, u_i, g^{f(j)}, v_{i,j})$ が DH-tuple であることを証明する．

（2） $v_{i,j}$ の $k+1$ 個の値のどんな部分集合 R からでも，各サーバは $s = \sum_{j \in R} L_j f(j)$ のラグランジェ補完のテクニックを使い $(u_i)^s$ を計算する．ここで，L_j は集合 R による適切なラグランジェ係数である．

（3） 各サーバは $e'_i = e_i (u_i)^s$ を計算する．

2.3.5 評　　価

公開鍵暗号方式を利用しているため再暗号化のコストは非常に高いが，この様な考えの下に，システム管理者の権限を持った者でも秘密鍵を知り得ることなしに安全に再暗号化をすることが可能である．また，仮に鍵が漏洩しても，その前後の鍵は推測すらできないため，復号されることは無い．

3. revocation

encrypt-on-disk 方式のシステムでは，アクセス権の失効したユーザ (revoked user) は再暗号化が完了するまでは復号のための鍵を所有していることになる．この鍵が漏洩すると，DaaS へのアクセスはできなくても，通信経路のデータを傍受すること等により復号が可能となり，機密情報の漏洩につながる．

このため，revocation が発生した場合に再暗号化は必須であるが，再暗号化を行うタイミングを考慮しなければならない．このタイミングにより revocation は active revocation, lazy revocation に分類される．

3.1 active revocation

revocation が発生した際，active revocation は直ちに新しい暗号鍵を生成し，対象データの再暗号化を実行する方式である．revoked user は新しい暗号鍵を知り得ないため，revocation 直後から復号が出来なくなり，情報漏洩を防ぐことが可能となる．

しかし，新しい鍵による再暗号化が終了するまで対象データを閲覧すらできなくなるため，再暗号化を直ちに行わねばならない．一般に，再暗号化の計算コストは高いため，セキュリティ面では優れているものの，revocation 時の性能の低下が問題となる．

3.2 lazy revocation

lazy revocation は，active revocation と異なり直ちに再暗号化を行わず，対象データの更新が発生するのを待って再暗号化を行う方式である．対象データの更新の際には暗号化処理も行うため，再暗号化の計算コストを削減することができる．このため，更新頻度が低いデータに関しては，active revocation が revocation 時と更新時の両方で暗号化処理を行うに対して，計算コストを低く抑えることができる．性能面では active revocation に比べて，非常に良い．

しかし，更新時まで再暗号化がされないため，revoked user が何らかの方法でアクセスが可能となった場合，本来復号できないデータを復号することが可能となる事態が生じる．このため，計算コスト面では優れているものの，セキュリティ面では問題がある．

3.3 BA-Rev

BA-Rev は，プライマリ・バックアップ構造を持つ分散ストレージシステムを利用する方式である．共通鍵暗号方式を利用し，プライマリデータは利用中の共通鍵で暗号化するが，図 3 に示したようにバックアップデータは次に利用する共通鍵で暗号化して格納する．次に利用する共通鍵は管理者以外には未知であるため，仮にバックアップデータにアクセスできたとしても復号することはできない．また，共通鍵は，ロックボックスに公開鍵暗号方式を用いて暗号化されて格納される．

revocation 時にはバックアップデータをプライマリデータに切り替えることにより，直ちにアクセス可能となる．active revocation が再暗号化が終了するまでアクセスできないという欠点と，lazy revocation が更新が終了するまで前の暗号鍵で復号できてしまうという欠点の，双方の欠点を補う方式である．

但し，データの更新時のコストが高いため，双方の方式よりもデータ更新時の性能に問題がある．また，revocation 時には，バックアップデータがプライマリデータに切り替わるため，バックアップデータが一時的に無くなってしまう．このため，active revocation と同様に，一度にバックアップデータを作成しなければならない．尚，この時に作成しているのはバックアップデータなので，プライマリデータにアクセスできなくなる訳ではない．

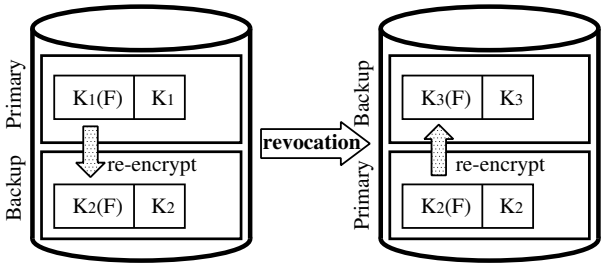

図 3　BA-Rev

4. 提案するシステム

以上により，本稿では DaaS を実現するために，再暗号化のために 2.3 に示した手法を用い，revocation には 3.3 に示した BA-Rev を用いることとする．

再暗号化において採用した手法では，管理者の権限を持った者が暗号文を取得したとしても，秘密鍵は知り得ないため安全に再暗号化を依頼することが可能である．また，仮にある期間の鍵が漏洩したとしても，その前後の鍵は安全であるため継続して利用することが可能である．但し，当該手法では公開鍵暗号方式を用いており計算のコストは非常に高い．しかし，セキュリティ面で非常に高い評価を与えることが出来るため，性能を犠牲にしてもシステムとしての安全性を優先する価値はあると考える．

また，revocation 時の対応としては，BA-Rev を取り入れた．やはり active revocation のようにアクセスできない状況が生じる性能の問題，lazy revocation のように古い鍵で復号できる状況が生じるセキュリティの問題はいずれも避けなければならない問題と考えたからである．但し，こちらも性能面において，更新時に関する計算コストが高くなっていた．これはロックボックスの利用に加えて，現在の共通鍵と次の共通鍵の双方で暗号化を行っているためであった．しかし，2.3 で示した通り，バックアップデータとして格納する新しい鍵で復号可能なデータの生成には，BA-Rev が提案された共通鍵を利用する環境ほどのコストは比率的に増えないと考える．これは，共通鍵を利用する場合は暗号化と復号を行わねばならないことに対して，本稿で提案したものは復号や暗号化の作業を行わず直接再暗号化を行うためである．更に，revocation の際は，バックアップデータをプライマリデータに変更した後，新たにバックアップデータを作成しなければならない．この際も同様に，新しいプライマリデータから直接再暗号化をすることが出来るため，BA-Rev が提案された共通鍵を利用する環境ほどコストは比率的に増えない．

再暗号化において公開鍵暗号方式を採用することにより性能面で大きな犠牲を払うことになるが，逆に revocation で採用した BA-Rev のデータ更新時の負荷が改善されることになる．

5. まとめと今後の課題

本稿では，再暗号化に 2.3 に示した手法を用いているため，暗号方式には公開鍵暗号方式を用いている．公開鍵暗号方式は計算コストが非常に高いため大規模なデータベースを対象とすることはできないが，セキュリティ面でシステムとしてより堅牢さを要求される場合には有効な選択であると考えられる．

DaaS の普及に伴い，DBMS の構築, 管理, 運用の人的コストを金銭的なコストに変換することが可能となっている．一方で，DB の規模に関わらず，そこに蓄えられるデータの重要度には変わりがない筈である．このため，構築したとしても DB が大規模とはならないが利用を敬遠してきた人々が，今後利用者として参入してくると考えるられる．本稿で提案した方式では，DB が大規模になればなるほど性能面で問題になると思われるが，DB が大規模とはならない利用者にとってはセキュリティの観点からも利用価値が高いと思われる．

本稿では，DaaS に再暗号化を依頼できるモデルを示したが，実装までには至っていない．再暗号化の計算コストを実際のデータで計測し，どの程度のデータ量であれば実用に耐えうるかを明確にすることも重要な検討課題であると考える．

また，本稿で採用した再暗号化の手法は拡張 ElGamal 暗号方式で実現できるが，それ以外の公開鍵暗号でも利用できることを確認し利用できる暗号系を増やすことにより，システムとしての可能性を広げることも必要であると考える．

文　献

[1] R. Canetti, S. Halevi and J.Katz, ”A Forward-Secure Public-Key Encryption Scheme”, Eurocrypt 2003, Lecture Note in Computer Science Vol.2656, Springer Verlag, pp.255–271, 2003

[2] E.J. Goh, H. Shacham, N. Modadugu, and D. Boneh, ”SiRiUS: Secure Remote Untrusted Sstorage”, the Proceedings of Interne Society(ISOC) Network and Distributred System Security (NDSS) Symposium 2003.

[3] M. Kallahalla, E. Riedel, R. Swaminathan, Q. Wang, and K. Fu, ”Plutus: Scalable Secure File Sharing on Untrusted Storage”, Fast ’03: Proceedings of the 1st USENIX Conference on File and Storage Technologies, pp.29–42, USENIX Association, 2003.

[4] E. Miller, D. Long, W. Freeman, and B. Reed, ”Strong Security for Network-Attached Storage”, Fast ’02: Proceedings of the 1st USENIX Conference on File and Storage Technologies, pp.1–13, Berkeley, CA, USA, 2002, USENIX Association.

[5] 尾形わかは, 黒澤 馨, Duong Quang Viet, 野口 宏, ”暗号化データ保管サーバシステム，暗号化データ保管方法及び再暗号化方法”, 特願 2004-056613, 2004

[6] 高山 一樹, 横田 治夫, ”暗号化データの複製を利用した権限失効処理手法の評価”, DEIM 2009 E1-3, 2009

[7] 高山 一樹, 横田 治夫, ”平文を生成しない分散ストレージでの再暗号化手法”, DBSJ Vol.7 No.3, 12, 2008