社団法人 情報処理学会 研究報告
IPSJ SIG Technical Report

2008－DBS－146 (29)
2008／9／22

平文を生成しない分散ストレージ上での再暗号化手法の提案

高山 一樹† 横田 治夫††,†

† 東京工業大学 大学院 情報理工学研究科 計算工学専攻 〒 152–8552 東京都目黒区大岡山 2–12–1 W8-82
†† 東京工業大学 学術国際情報センター 〒 152–8552 東京都目黒区大岡山 2–12–1
E-mail: †takayama@de.cs.titech.ac.jp, ††yokota@cs.titech.ac.jp

あらまし 情報セキュリティの重要性が増大する中で，並列ストレージにおける伝送中データを保護する手法として，データを暗号化して格納しておく encrypt-on-disk 方式がある．この方式はデータ転送性能に優れる半面，暗号鍵の更新が必要な場合の処理コストが大きい．encrypt-on-disk 方式の高機能ストレージにおいて，再暗号化処理をディスク上で行うことでクライアント側の負担を軽減することができるが，通常の再暗号化手法では一時的に平文が生成される為，情報漏洩の可能性がある．我々は分散ストレージの構造と特定の暗号化方式の性質を利用した，平文を生成しない再暗号化手法を提案する．OFB モードと排他的論理和の性質に着目し，暗号化と復号を逆順に処理することで，平文の生成を防ぐ．本稿で性能面とセキュリティ面で評価し，有用性を示す．

キーワード 暗号化，高信頼ストレージ，セキュアストレージ，encrypt-on-disk

A Re-encryption Method on a Distributed Storage without Generating Cleartexts

Kazuki TAKAYAMA† and Haruo YOKOTA††,†

† Department of Computer Science,Tokyo Institute of Technology Ookayama 2–12–1, Meguro-ku, Tokyo, 152–8552 Japan
†† Global Scientific Information and Computer Center, Tokyo Institute of Technology Ookayama 152–8552, Meguro-ku, 152–8552 Japan
E-mail: †takayama@de.cs.titech.ac.jp, ††yokota@cs.titech.ac.jp

Abstract The information security becomes more and more important recently. There is encrypt-on-disk scheme, in which files are stored in cipher, and it is a scheme for protection of transmitted data in parallel storage systems. This scheme costs re-encryptions with the updates of cryptographic keys, while it has improved performance of data transmission. With a high functional storage adopted encrypt-on-disk scheme, we can reduce the costs in clients by processing re-encryption on storage nodes. However, there is possibilities of leaks because cleartext is generated along normal re-encryption processes. We propose a re-encryption method without generating cleartexts by applying structure of distributed storage and particular encryption mode. We focus on characteristics of OFB mode and XOR. The cleartext generations are prevented by processing encryption and decryption in reverse. In this paper, we evaluate the proposal with performance and security, and indicate the usefulness of proposed method.

Key words encryption, high dependable storage, secure storage, encrypt-on-disk

1. まえがき

近年，情報セキュリティの重要性が増大する中，ネットワークストレージにおいても様々なデータ保護要件の考慮が必須となっている [1], [2]．保護要件の一つ，伝送中データの機密性保持の為の手法として，encrypt-on-disk 方式がある．これはデータを暗号化した状態でストレージに格納する方式で，転送時のみ暗号を用いる encrypt-on-wire 方式と比較して転送性能が良い反面，ユーザのアクセス権失効（revocation）に伴い，暗号化データを新しい鍵で再暗号化しなければならない．

一方，我々はこれまで，高機能な分散ストレージシステムとして，自律ディスク [3] を提案してきた．自律ディスクはネットワークに接続された高機能ディスクノードのクラスタにより構成される．ノードの演算処理能力を利用し，耐故障化，負荷均衡化，容量分散等の機能を自律的に実行する．

encrypt-on-disk 方式を採用した高機能ストレージでは，re-

vocation に伴う再暗号化処理をストレージノード上で行うことが望ましい．そうすることでクライアントの負担を軽減でき，同時に encrypt-on-disk 方式の目的である伝送路の機密性も維持できる．しかし一方で，通常の再暗号化手法では処理中に平文の状態がある，また再暗号化の為に獲得可能な形で暗号鍵が保存されている等の理由より，ディスクノード上の機密性は提供できない．

本稿では，分散ストレージにおける，単一ストレージノード上での機密性を実現できる再暗号化手法 RORE（Reverce Order Re-Encryption）を提案する．ブロック暗号の暗号化モードのうち，OFB （Output FeedBack）等一部のモードではその処理の性質上，再暗号化における暗号化処理と復号処理が可逆であるため，暗号化を先に処理することで平文を生成しない再暗号化処理を実現でき，ノード陥落によるデータ漏洩を防ぐことができる．

以下，本稿の構成を述べる．2. で encrypt-on-disk 方式について説明する．3. で関連研究について述べる．4. で，本研究の前提となるシステム要件について述べる．5. で，既存方式での問題点と，それに対する我々の提案を説明する．提案について，6. で性能面について，7. でセキュリティ面について考察を行う．最後に，8. で本稿のまとめと今後の課題について述べる．

2. encrypt-on-disk 方式

2.1 概要及び転送性能

encrypt-on-disk 方式は，ネットワークストレージにおける，伝送路上のデータを悪意あるユーザの傍受から守る暗号利用方式の 1 つである．encrypt-on-disk 方式ではデータを暗号化した状態でストレージノードに格納する．その為，ストレージ側でのデータ送受信時に暗号処理を必要としない．

他方式として encrypt-on-wire 方式が挙げられる．この方式ではデータを平文の状態で格納し，送受信時に暗号化及び復号処理を行う．

この二方式をデータ転送性能の面で比較すると，encrypt-on-disk 方式の方が優れている．これは encrypt-on-wire 方式ではストレージ側送受信時に必ず暗号処理が発生すること，セッション毎に新しい暗号鍵を生成するため，鍵生成コストがかかることが理由である．

2.2 encrypt-on-disk におけるアクセス権失効

複数のユーザでデータを共有してるシステムの場合，encrypt-on-disk 方式ではユーザのアクセス権失効（revocation）に伴い，対象データを新しい暗号鍵で再暗号化する必要がある．これは，アクセス権を失効されたユーザ（revoked user）は現在使われている暗号鍵を保持している可能性があり，アクセス制御リスト（ACL）等のアクセス管理手法で revoked user のアクセス要求を拒否しても，傍受等不当なアクセスで revoked user にデータが渡ると，情報が漏洩してしまうからである．

この為 revocation 処理に関しては encrypt-on-disk 方式は encrypt-on-wire 方式より処理コストが高い．しかし，revocation コストを考慮した上でも，総合して encrypt-on-disk システムは encrypt-on-wire システムより性能面，セキュリティ面共に優れていると検証されている [1].

3. 関 連 研 究

encyrpt-on-disk 方式を採用したセキュアストレージシステムとして，SNAD [4]，Plutus [5]，SiRiUS [6] 等がある．これらのシステムはサーバが不正を行わないことを信用できない（trust でない）という前提で，データが平文として存在し得るのはクライアントマシン上のみであるとしている．また，revocation 処理は所有者主導で実行される．我々の研究ではクライアントの負担を軽減するために，ストレージ側で再暗号化処理を行い，かつ後述の通り処理とデータ配置の考慮によりノード上のセキュリティ実現を目指すため，方針が異なる．

我々は分散ストレージにおける，性能とセキュリティを両立した revocation 時再暗号化手法，BA-Rev（Backup Assist Revocation）[7] を提案した．予めバックアップデータを新しい鍵で暗号化しておき，revocation 発生時にプライマリとすることで，アクセス回復までの時間を再暗号化処理の分短縮することができる．次に元のプライマリをバックグラウンドで再暗号化し，新しいバックアップとして設定する．このように再暗号化処理をストレージノード上で処理する点は本稿の方針と同じであり，また本稿の提案手法はこの BA-Rev への適用を想定するものとする。

4. 想定するシステム要件

以下本稿で前提とするシステムやデータ構造について概略を述べる．

4.1 高機能分散ストレージシステム

ストレージ装置上の演算処理能力を利用してデータの管理を自律的に行うシステムとして自律ディスク [3] を提案してきた．自律ディスクはネットワークに接続された高機能ディスクノードのクラスタにより構成される．この高機能ディスクの演算能力を利用し，ストレージ側で耐故障化，負荷均衡化，容量分散等の機能を自律的に実行し，ユーザによるストレージ管理の負担を軽減する．

本稿では，encrypt-on-disk 方式を採用した，自律ディスクのような高機能ストレージシステムや，ファイルサーバクラスタを対象とする．自律ディスクの方針に従い，revocation に伴う再暗号化処理等はストレージシステム側で行い，クライアント側の負担を軽減する．

4.2 ネットワーク構成

各ストレージノードは並列にネットワーク接続される．これらのストレージノードに対し，同様にネットワークに接続されたクライアントノードからアクセスするものとする．

4.3 暗号の利用

本研究で利用した暗号の種類と暗号化モード，鍵管理方法について説明する．

4.3.1 暗号の種類と利用方法

暗号は，暗号化と復号に共通の鍵を用いる共通鍵暗号と，異なる対の鍵を用いる公開鍵暗号に分類できる．共通鍵暗号は，予め暗号化側と復号側で，安全な方法で鍵を共有していなけれ

Key Object ID	User ID	Signature	Reference Count
User ID	Encrypted key		Permissions
A	$K_A^+(K)$		P_A
B	$K_B^+(K)$		P_B

K_A^+ :A's public key

K :secret key

図 1　ロックボックスの例 : *key object*

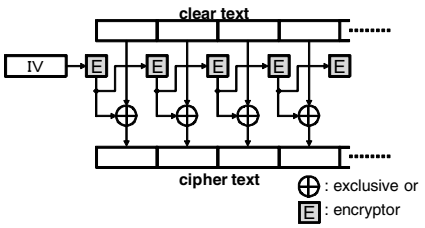

図 2　OFB モード

ばならない．一方，公開鍵と秘密鍵の対を用いる公開鍵暗号では，対の内の片方（公開鍵）を公開することができる為，鍵配布の問題はない．しかし，その反面，公開鍵暗号は一般的に共通鍵暗号の数百から数千倍の処理速度であるという問題がある。

これらの特性を考慮し，一般的には共通鍵の配布を公開鍵暗号を用いて行い，その共通鍵を用いてデータのやりとりをする場合が多い．本研究においてもこの方式をとる．

4.3.2　鍵管理構造

データの暗号鍵を管理する構造としてロックボックスがある．ロックボックスとは，暗号鍵を格納し，アクセス権を持つユーザのみ鍵を取り出すことができる鍵管理構造である．SNAD [4] におけるロックボックスである *key object* の例を図 1 に示す．ここでユーザ x は公開鍵 K_x^+ と秘密鍵 K_x^- を持ち，あるデータは共通鍵 K で暗号化されているものとする．*key object* の各行はユーザ ID，ユーザの公開鍵で暗号化された共通鍵 $K_x^+(K)$，ユーザに認可されているアクセス権の種類 P_x からなる．格納された共通鍵を獲得するには，公開鍵と対を成す，ユーザ自身のクライアントノード上に保存された秘密鍵を用いてのみ復号できる為，正しく認可されたユーザのみが共通鍵を獲得できる．

本研究では，この *key object* の構造を利用して鍵を管理する．1 ファイルに対し 1 つの *key object* が対応し，ファイルを暗号化した鍵を格納し，ストレージノードに格納される．また本研究ではストレージ側で再暗号化等の暗号処理を行う為，各ストレージノードもユーザと同様に公開鍵と秘密鍵を持ち，処理対象のファイルの *key object* に獲得可能な鍵を保持するものとする．

4.3.3　OFB 暗号化モード

OFB（Output FeedBack）モードは，ブロック暗号の暗号化モードの一つである．本研究ではこの暗号化モードを使用した．処理の流れの簡略図を図 2 に示す．

暗号文の最初のブロックは初期ベクトル（Initial Vector:IV）を暗号化して疑似乱数ビット列を生成し，平文のブロックとの排他的論理和演算で得られる．後続のブロックは，前ブロックの疑似乱数ビット列をさらに暗号化し，同様に平文との排他的論理和演算をすることで得られる．

OFB モードの特色として，暗号処理部分を平文とは独立に実行でき，実際の平文への処理は排他的論理和のみである点，ストリーム暗号的に利用される点が挙げられる．また暗号化と復号の処理が同じであるのも特徴である．

OFB モードは初期ベクトルの暗号化に始まり，先頭ブロックから順次処理しなければならない為，ランダムアクセスに弱いという欠点がある．これに対し，CTR （counter）モードというランダムアクセスに強いモードがある．CTR では暗号化処理の入力として，前ブロックの疑似乱数ビット列ではなく，ブロックの位置に対応した値を持つカウンタの値を入力として用いるため，途中のブロックからの処理が可能である．

4.4　想定する攻撃モデル

攻撃者は，対象となる分散ストレージシステムと同じネットワーク上に存在し，次の攻撃を行えるものとする．

- 伝送路へ攻撃し，転送中のデータを傍受する．
- あるストレージノードに攻撃し，陥落させて内部のデータを奪取する．

なお，攻撃者は格納されたファイルへのアクセス権は持たず，鍵等いかなるデータも所持していないものとする．

5.　RORE の提案

自律ディスク [3] 等の高機能ストレージシステムは，様々な処理をストレージシステム側で自律的に処理することで，ユーザやストレージ管理者の負担を軽減する．伝送データ保護方式の中ではデータ転送性能が良い encrypt-on-disk 方式を高機能ストレージに適用する場合，revocation 時の再暗号化処理もストレージ側で処理することで，同様にクライアントの負担を軽減することが可能である．

しかしその一方で，通常の再暗号化処理は暗号化データを復号し，その後暗号化する為，中間生成物として平文が現れるので，ストレージノード上の機密性は守られないという問題があった．

この問題点に対し，処理中に平文が生成されない再暗号化手法 RORE（Reverse Order Re-Encryption）を提案する．

4.3.3 で説明した OFB モードでは，暗号化は平文と疑似乱数ビット列の排他的論理和を求めることで実行される．また復号は暗号文に対して同様の処理を行うことで実行される．ここで，排他的論理和は交換則が成り立つ為，再暗号化処理における暗号化と復号も可逆である．この性質を利用し，再暗号化処理を“暗号化”，“復号”の順に処理することで，平文を生成しない再暗号化を実現でき，再暗号化処理中の攻撃者による当該ノード陥落によるデータ漏洩を防ぐことが可能となる．

なお，定常状態を含めて，1 ノード陥落による漏洩を防ぐ為

には，鍵管理の考慮が必要である．再暗号化をストレージ側で行う為に，共通鍵をストレージノードが利用可能な形で保存する必要があるが，この無防備な状態の鍵を暗号化ファイルと同じノードに置くと，そのノードが陥落した時点でデータが漏洩してしまうからである．対策としては，後述の他ノードで管理する方法や，暗号化を専用のハードウェアを用いて行うことで鍵を奪取不可能にする方法等が考えられる．後者の例としては，HDD に暗号化用回路を組み込んだ Seagate の DriveTrust [8] や富士通の MTZ2 CJ がある．

6. 実　　験

提案手法の性能面での評価の為，PC クラスタ上で動作する encrypt-on-disk 方式のファイルサーバ・クライアントプログラムを作成し，実験を行った．

6.1 実験プログラム

実験プログラムは以下の 2 環境について，ファイル獲得（get）の応答時間と，ストレージ上再暗号化処理の実行時間を測定できる．

i. **normal** : 通常の再暗号化処理を行う環境
復号，暗号化の順に再暗号化処理を行う．ロックボックスは対象の暗号化ファイルと同じストレージノードに格納する．

ii. **RORE** : 再暗号化で RORE を適用した環境
5. で述べた通り，ロックボックスは図 4 の通り隣接ノードに置くものとする．

暗号の利用方法や鍵管理は 4. の記述に従う．ここでは，提案の環境では，鍵を管理するロックボックスを対応する暗号化ファイルとは異なるノードに置くことで，1 ノード陥落によるデータ漏洩を防ぐものとする．

図 3，4 に各環境での再暗号化処理の概略を示す．提案手法である RORE における処理の流れは以下の通りである．ここでの初期状態として，ファイル F はユニークな鍵 K_1，OFB モードで暗号化されてノード A に格納され，そのロックボックスは隣接ノード B に格納されている．ロックボックスには再暗号化処理用に，ロックボックス格納ノード N の公開鍵 K_N^+ で暗号化された K_1 を保存している．なお，このプログラムにより，この手順で処理を行うことで実際に平文を生成せず再暗号化処理を完了できることを確認できた．

step 1　暗号化ファイルを格納したノードで新しい鍵 K_2 を生成，暗号化する．この段階でファイルは二重に暗号化された状態 $K_1K_2(F)$ となる．

step 2　ノード B のロックボックスに K_2 を転送し，代わりに K_1 を受信する．この時鍵交換は送信先ノードの公開鍵で暗号化した状態で行う為，送信前後及びロックボックスの再構成時に復号，暗号化の必要がある．

step 3　古い鍵 K_1 でファイルの復号を行い，その後 K_1 を破棄する．

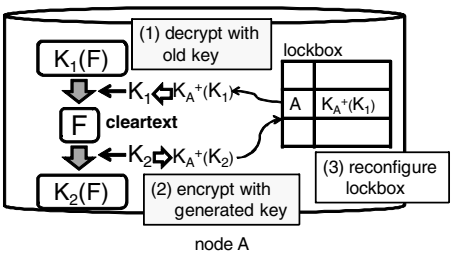

図 3　通常の再暗号化手法の処理

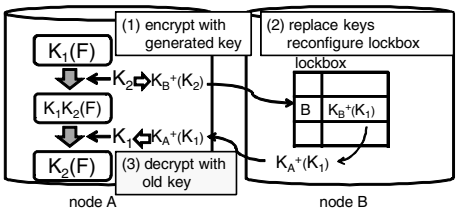

図 4　RORE での再暗号化手法の処理

表 1　ストレージノード諸元

CPU	AMD Athlon XP-M1800+ (1.53GHz)
Memory	PC2100 DDR SDRAM 1GB
HDD	TOSHIBA MK3019GAX (30GB, 5400rpm, 2.5inch)
Network	TCP/IP + 1000BASE-T
OS	Linux 2.4.20
Java VM	Sun J2SE SDK 1.5.0_03 Server VM

表 2　固定パラメタ

公開鍵	RSA 1024bit
共通鍵	AES 128bit
暗号化モード	OFB
パディング	NoPadding
ストレージノード数	3
ノード当たりデータサイズ	1MB×500
Zipf 母数 θ	0.7

プログラムで実行できる処理のうち，get はロックボックスに格納された対象ユーザ用の共通鍵の転送も含む．その為，ロックボックスの格納場所が異なる 2 環境では，再暗号化処理だけでなく get の処理も流れが異なるため，以下評価の対象とする．

6.2 実 験 環 境

実験は表 1 に示す構成の PC クラスタ上で行った．またパラメタとして表 2 のものを用いた．

表の記述の通り，実験では 3 台のストレージノードに其々 1MB，500 個のファイルを暗号化して格納する．ただし総データサイズにロックボックスのサイズは含まない．

また，get の対象となるファイルは，選択したストレージノードの全ファイルの中から，パラメタ θ によって決まる Zipf 分布 [9] に従い偏って選ばれるものとした．

6.3 実験 1 : get の応答時間の比較

ファイルを格納した 3 ノードに対し，それぞれ 1 つのクライアントノードから get リクエストを 1000 回実行し，その応答

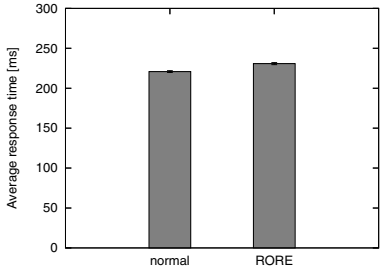

図 5　get コマンドにおける平均応答時間の比較

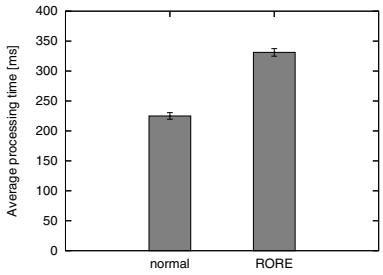

図 6　再暗号化の処理時間の比較

時間を測定した．図 5 は各環境における，3 ノード分の平均応答時間，及び 95%信頼区間を表している．

実験の結果，RORE を適用した環境では get の応答時間が平均で約 4.5 パーセント増加した．これは，提案の環境では，ファイル格納ノードが隣接ノードからアクセスしたユーザに対応する共通鍵を取得しなければならない為だと考える．ただし鍵は小さい固定長である為，その転送の影響は非常に小さいことがわかる．

6.4　実験 2：再暗号化の処理時間の比較

ファイルを格納した 3 ノード上で，それぞれが再暗号化処理を 1000 回実行し，その実行開始から終了までの時間を測定した．ここで再暗号化対象のファイルはランダムで選択するものとする．図 6 はそれぞれの環境における処理時間の平均と 95%信頼区間を表している．

図より，RORE の再暗号化処理時間は，通常に比べて約 100 ミリ秒，約 47 パーセント増加した．これはノード間通信が必要であることに加え，共通鍵転送時に公開鍵暗号による暗号処理が必要である為だと考える．ここでは，ノード A から B への新しい鍵の送受信時，新しい鍵のロックボックスへの設定時，B から A への古い鍵の送受信時に公開鍵暗号での暗号処理が必要となる．6.3 の結果より，鍵の転送処理自体の影響は非常に小さいことより，公開鍵暗号による処理が性能に与える影響は大きいことがわかる．

6.5　実験 3：再暗号化処理が他アクセスに与える影響の比較

再暗号化処理が，同時に発生した他のアクセスに対してどの程度の影響を与えるかを実験により測定した．ノード A，B にファイルを格納した状態で，ノード A で再暗号化処理を連続で実行する．この時ノード B は，提案手法の環境において，ノード A に格納されたファイルのロックボックスが格納されるノードである．すなわち，normal ではノード A のみで再暗号化処理が実行され，RORE では図 4 の通りノード A で再暗号化処理，ノード B でロックボックスの処理が実行される状態である．この状態でノード A，B に対し get を 1000 回実行し，応答時間を測定した．図 7 は各環境，各ノードに対する get の平均応答時間，95%信頼区間を表す．

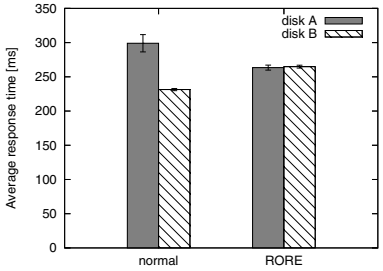

図 7　バックグラウンドの再暗号化処理が get に与える影響の評価

結果は，再暗号化処理の無い平常状態（図の normal:ノード B）と比較して，get の応答時間は通常の環境で約 29 パーセント，RORE ではノード A，B 共に約 14 パーセント増加した．通常では再暗号化処理が 1 ノード内で行われる分，大きく他のアクセスに影響を与えるのに対し，処理が 2 ノードに分散される RORE では，他アクセスへの影響も分散されることがわかる．

6.6　性能に関する考察

6.3 の結果より，ファイル本体とロックボックスの格納ノード分離による get の性能低下は非常に小さいことがわかり，無視できる程度であると考える．

6.4 より，再暗号化の処理時間は増加する．その為，再暗号化の発生箇所とタイミングによっては性能低下を引き起こす可能性がある．例えばアクセス対象のファイルが再暗号化中の場合，処理が終わるまでアクセスがブロックされ，応答時間が大きくなってしまう．

しかし一方で，本研究は 3. で述べた revocation 時再暗号化手法である BA-Rev [7] への適用を考えており，再暗号化処理時間増大の影響は小さいと考える．BA-Rev では revocation 発生後に使われるデータを予めバックアップに保持しており，そちらにアクセスを回送し，実際の再暗号化処理はバックグラウンドで行われる．その為再暗号化処理中のファイルが直接アクセスされることはないからである．

バックグラウンドでの再暗号化処理による性能への影響は，6.5 の結果より，RORE では複数ノードで影響が出る反面，1 ノード当たりの性能劣化度合は小さい．QoS（Quality of Service）の観点から見ると，突出して性能劣化するノードがない為，優れていると言える．なお，実際に BA-Rev に適用した場合の性能評価は今後の課題とする．

7. セキュリティ面に関する考察

6. の環境では，RORE を適用することにより，5. で述べた通り，再暗号化処理中に平文が生成されず，また 1 ノードに平文生成が可能となるデータ（暗号化ファイル，共通鍵）が同時に存在する期間がない為，1 ノード陥落時のデータ漏洩を防ぐことができる．また，あるファイルに関する，ファイル格納ノードと対応するロックボックス格納ノードのうち，何れかが陥落しない限り，データ漏洩は防ぐことができる．一方で，あるファイルの格納ノードとロックボックス格納ノードが同時に陥落した場合，或いはロックボックス格納ノードが陥落し，かつ通信中の対象ファイルが傍受された場合，そのファイルは漏洩する．しかし，再暗号化処理で平文が生成されてしまう通常の方式のように，1 ノード陥落でデータが漏洩してしまうシステムと比較すると，分散ストレージシステムの構造上，提案方式では攻撃者が複数ステップを踏まなければデータが漏洩することがない為，提案方式の環境の方がセキュリティ面で優れているといえる．

また，3. で挙げた既存のセキュアストレージシステムのようにシステムを trust でないと仮定した環境と比較すると，本稿の提案はストレージ側で再暗号化処理等を行うために生の鍵を扱っている分，セキュアでないといえる．しかしその反面，再暗号化処理をユーザの手を煩わせずストレージ側で行うことで，クライアント側の負担を大きく軽減することが可能となっている．その為，RORE を適用した高機能ストレージシステムは，多人数で共有する大容量の分散ファイルサーバ等への適用が適している．このようなシステムではデータ管理コストが大きく，ストレージ主導で管理を行い，ユーザ側の負担を軽減することが望ましい．RORE 及び BA-Rev を適用することで，revocation 時のクライアント側の負担及びシステム性能劣化を防ぎ，かつ通常の再暗号化処理を行うよりも高い機密性を維持することができる．

8. まとめと今後の課題

分散ストレージにおける，単一ストレージノード上での機密性を実現できる再暗号化手法 RORE を提案した．再暗号化処理中の暗号化処理と復号処理を可逆とできる OFB 暗号化モードを利用することで，途中生成物として平文を生成しない再暗号化処理を実現でき，再暗号化処理時のノード陥落による漏洩を防ぐことができる．それに加え，暗号化ファイルの保存と鍵の管理を異なるノードで管理する，或いは暗号処理専用のハードウェアをノードに搭載し鍵を奪取できないようにする等，1 ノード内のデータで平文が奪取可能な期間が無いように処理を考慮することで，上記の 1 ノード陥落に対する機密性が実現可能となる．

実験により RORE の性能面の評価を行った．ファイルとその鍵のロックボックスを異なるノードに格納した提案の環境における，ファイル獲得（get）の応答時間を測定したところ，通常の環境との差は小さかった．一方，再暗号化の処理時間は通常より 47 パーセント増加したものの，バックグラウンドでの再暗号化処理が他のアクセス（ここでは get）の応答時間に与える影響は，通常は 1 ノードで 29 パーセント増だったのに対し，提案方式では 2 ノードそれぞれで 14 パーセント増と分散された．その為 BA-Rev [7] のようにアクセスされないデータのみ再暗号化されるような手法へは性能を低下させずに適用できると考える．また通常の再暗号化処理のように，突出して性能が劣化するノードの出現を防ぐことができ，QoS の観点から見ると提案方式は優れていると言える．

今後の課題としては，本稿で提案した RORE を，revocation 等，ユーザにより実際に要求されるコマンドの処理に組み込み，評価することが挙げられる．その中で，我々が過去に提案した BA-Rev にも組み込み，性能面，セキュリティ面双方から評価を行うことが必要である．

謝　　辞

本研究の一部は，独立行政法人科学技術振興機構戦略的創造研究推進事業 CREST，および文部科学省科学研究費補助金特定領域研究 (19024028) の助成により行なわれた．

文　　献

[1] E. Riedel, M. Kallahalla, and R. Swaminathan, “A framework for evalating storage system security,” FAST ’02: Proceedings of the 1st USENIX Conference on File and Storage Technologies, pp.15–30, USENIX Association, 2002.

[2] P. Stanton, “Securing Data in Storage: A Review of Current Reserch,” ArXiv Computer Science e-prints, 2004.

[3] H. Yokota, “Autonomous Disks for Advanced Database Applications,” Proc. of International Symposium on Database Applications in Non-Traditional Environments (DANTE’99), pp.435–442, Nov. 1999.

[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, p.1, Berkeley, CA, USA, 2002, USENIX Association.

[5] 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.

[6] E.J. Goh, H. Shacham, N. Modadugu, and D. Boneh, “SiRiUS: Securing Remote Untrusted Storage,” the proceedings of the Internet Society (ISOC) Network and Distributed Systems Security (NDSS) Symposium 2003, 2003.

[7] 高山一樹，小林大，横田治夫，“複製を利用したストレージ中での暗号化データの権限失効処理，” 第 18 回データ工学ワークショップ（DEWS2007）予稿集，電子情報通信学会データ工学専門委員会，2/3 2007.

[8] Seagate, “Drivetrust^{tm} technology: A technical overview,”, http://www.seagate.com/docs/pdf/whitepaper/TP564_DriveTrust_Oct06.pdf.

[9] D.E. Knuth, Sorting and Searching, Addison-Wesley Publishing Company, 1973.