論文 / 著書情報
Article / Book Information

論題(和文)	暗号化データ格納ストレージにおける性能とセキュリティの両立
Title(English)	Performance and Security on Storage System with Encrypted Data
著者(和文)	高山一樹, 横田 治夫
Authors(English)	Kazuki TAKAYAMAy, Haruo YOKOTA
出典(和文)	電子情報通信学会和文論文誌D, J93-D, 3, 241-252
Citation(English)	IEICE Transactions on Information and Systems, J93-D, 3, 241-252
発行日 / Pub. date	2010, 3
URL	http://search.ieice.org/
権利情報 / Copyright	本著作物の著作権は電子情報通信学会に帰属します。 Copyright (c) 2010 Institute of Electronics, Information and Communication Engineers.

暗号化データ格納ストレージにおける性能とセキュリティの両立*

高山 一樹† 横田 治夫††,†

Performance and Security on Storage System with Encrypted Data*

Kazuki TAKAYAMA† and Haruo YOKOTA††,†

あらまし ネットワークストレージにおけるセキュリティと性能の両立は極めて重要な課題である．セキュリティを保つためにデータを暗号化して格納するネットワークストレージにおいて，一部利用者のアクセス権が失効した場合，再暗号化が必要となる．これまでの手法では，セキュリティを十分確保するために再暗号化の間アクセスができずサービスの質を確保できなかった．我々は複製をもつ暗号化データ格納ストレージにおける再暗号化手法として BA-Rev を提案している．BA-Rev は複製データをあらかじめ異なる暗号鍵で暗号化し，失効処理時に主データと切り換えることで，信頼性を確保しつつアクセス権失効時のアクセス不能期間を短縮し，処理コストも低減する．この BA-Rev の単純な実装では，既存方式の active revocation と同等のセキュリティを維持しながら読出し性能が向上できる反面，更新性能が悪化するという問題点がある．BA-Rev の更新性能を改善するため，複製側での書込みと再暗号化処理を遅延させる DW/DRW 戦略を BA-Rev に適用する．PC クラスタを用いた実験により，提案方式は既存方式と同等の更新性能を維持しつつ，権限失効時の処理の性能も向上することを示す．また，信頼性の見積りにより遅延による信頼性低下がほとんどないことを示す．

キーワード 分散ストレージ，セキュリティ，アクセス権失効，プライマリ–バックアップ構造，信頼性

1. まえがき

多数のクライアントからアクセスを受け付けるネットワークストレージにおいて，セキュリティと性能の両立は極めて重要な課題である [1], [2]．ネットワークストレージにおけるデータ保護要件の一つとして，ネットワーク伝送中データの機密性保護があり，encrypt-on-disk 方式が提案されている．これはデータを暗号化された状態でストレージに格納する方式で，転送時のみ暗号を利用する encrypt-on-wire 方式と比較して転送性能が高く，ノード内のデータの機密性も実現可能である．しかしその反面，利用者のアクセス権失効 (revocation) に伴い，新しい暗号鍵で対象データを再暗号化する必要がある．この権限失効時再暗号化処理の既存方式である active revocation と lazy revocation には，再暗号化を直ちに実行することによる性能低下と，再暗号化を遅延することによる脆弱性という，相反する欠点が存在する．つまり，セキュリティを十分確保するためにはアクセス不能期間が発生し，サービスの質を保つことができなかった．

我々は，active revocation と同等の安全性を保ちつつ，効率の良い再暗号化処理を実現する BA-Rev (Backup Assisted Revocation) を提案している [3]．BA-Rev はプライマリバックアップ構造を前提とし，バックアップデータをあらかじめ新しい暗号鍵で暗号化しておき，権限失効時にプライマリデータと置き換えることで，信頼性を確保しつつ，迅速かつ低コストな権限失効を実現する．ただし，BA-Rev では，プライマリだけでなくバックアップの更新も必要となるため，単純な実装では更新性能が悪化するという問題点がある．

この問題点を解決するため，BA-Rev の環境に DW(Delayed Writing) 戦略 [4] 及び DRW(Delayed Re-encrypting and Writing) 戦略 [5] を適用し，更新性能と権限失効処理時の性能に与える影響を評価する．DW 戦略は更新処理時にバックアップ側の差分データの書込みを，DRW 戦略は再暗号化及び書込みを遅延

† 東京工業大学大学院情報理工学研究科計算工学専攻，東京都
Department of Computer Science, Graduate School of Information Science and Engineering, Tokyo Institute of Technology, 2–12–1 Ookayama, Meguro-ku, Tokyo, 152–8552 Japan
†† 東京工業大学学術国際情報センター，東京都
Global Scientific Information and Computing Center, Tokyo Institute of Technology, 2–12–1 Ookayama, Meguro-ku, Tokyo, 152–8550 Japan
* 本論文はデータ工学研究専門委員会推薦論文である．

することで，負荷分散と複数回の更新に対する適用処理を集約し，性能向上を図る．これらの提案方式と，既存方式である active revocation の性能及び信頼性を比較し，既存方式に対する優位性を示す．

これらの提案手法は，プライマリバックアップ構造のように複数ノードにデータの複製をもつネットワークストレージにおいて，暗号化データを格納して扱うという条件を満たすシステムであれば適用可能である．ストレージシステムによく用いられる RAID の場合には，パリティ計算を行う RAID 3-6 構成内部に提案手法をそのまま適用することはできないが，近年複数の RAID を組み合わせて使うことも増えており，1 部の RAID をプライマリとし，残りをバックアップするような構成の場合には同様に適用可能である．

すなわち，本論文ではストレージ側でデータ管理処理を行う自律ディスク [6] の概念や，ノード上で共通鍵を管理するロックボックス [7] を利用しているが，それ以外の環境，例えば SAN に接続されサーバが管理するプライマリバックアップ構造のような環境であっても，暗号化データを格納する場合には，active revocation の安全性を維持しながら性能を向上させるという提案手法の効果は有効である．今後のストレージシステムにおいては，各ストレージノードを高機能化して，分散処理させることがスケーラビリティや信頼性の面から重要であるという観点から，本論文では各ストレージノード上での処理を前提としている．

以下に本論文の構成を述べる．**2.** で暗号化データを格納する encrypt-on-disk 方式の特徴を説明する．**3.** で関連研究について述べる．**4.** で本研究が前提とするシステムの概要を述べる．**5.** で BA-Rev の概要と DW/DRW 戦略を説明する．**6.** で実験とその結果に対する考察を行う．**7.** で，MTTDL の比較による信頼性の評価を行い，最後に **8.** で本論文のまとめと今後の課題について述べる．

2. 暗号化データ格納ストレージ

分散ストレージにおける暗号利用方式の一つである encrypt-on-disk 方式の特徴，利点及び問題点について説明する．

2.1 encrypt-on-disk 方式と encrypt-on-wire 方式

分散ストレージでは各ノード間で転送されるデータの保護が必須である．伝送路上のデータの機密性保護のための暗号利用方式として，encrypt-on-wire 方式と encrypt-on-disk 方式がある．encrypt-on-wire 方式は，セッションごとに新しく暗号鍵を生成し，それを用いてデータを暗号化し，転送する方式である．一方，encrypt-on-disk 方式は，あらかじめデータを暗号化した状態でストレージノードに格納しておき，転送時はその暗号化データをそのまま転送する方式である．

二方式をデータ転送時におけるパフォーマンスに関して比較すると，encrypt-on-disk 方式ではストレージ側でのデータ送受信時に暗号化及び復号処理を行う必要がないため，encrypt-on-wire 方式よりも効率が良い．また encrypt-on-wire 方式ではセッションごとに暗号鍵生成コストが掛かり，性能面で問題がある．

またセキュリティ面で比較すると，伝送路上の機密性は二方式で同等だが，encrypt-on-disk 方式ではストレージノード上の機密性も実現が可能であるため有用である [1]．

2.2 アクセス権失効処理

複数ユーザでデータを共有する，encrypt-on-disk 方式を採るシステムでは，あるユーザのあるデータに対するアクセス権の失効 (revocation) に伴い，対象データを再暗号化する必要がある．これは，アクセス権を失ったユーザ (revoked user) が対象データに用いられている暗号鍵を保持している可能性があり，アクセス制御により権限を失効されたユーザのアクセス要求を拒否していても，傍受等の不正アクセスによりデータが権限を失効されたユーザに渡ると，情報が漏えいする危険性があるためである．

このため，権限失効処理に関しては，encrypt-on-disk 方式は encrypt-on-wire 方式より処理コストが高い．しかし，権限失効処理時の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，セキュリティ面ともに優れると検証されている [1]．

2.3 権限失効時再暗号化手法

encrypt-on-disk 方式での権限失効処理における再暗号化は，処理を行うタイミングによって active revocation と lazy revocation に分類できる．

2.3.1 active revocation

active revocation は，権限失効発生後直ちに，新しい暗号鍵を生成し，対象データの再暗号化処理を実行する方式である．権限を失効されたユーザは新しい暗号鍵をもたず，権限失効直後から対象データを復号できなくなるため，後述の lazy revocation と比較して

セキュリティ面で優れる．しかし一方で，直ちに再暗号化処理を実行しなければならない，再暗号化処理が終了するまで対象データにアクセスできない等の原因より，性能を低下させる可能性がある．これは権限失効の対象が複数データに及ぶ場合顕著である．

2.3.2 lazy revocation

lazy revocation は，対象データの再暗号化処理を次回の更新時まで遅延する方式である．Cepheus [8] で提案され，Plutus [9] 等で採用されている．暗号化データの更新処理は暗号化処理を伴うため，権限失効のための暗号化処理を兼ねることでコストを削減できる．また，更新頻度が低いデータでは，権限失効のたびに再暗号化を行う active revocation と比べ，複数回の権限失効に対する再暗号化処理をまとめることができるので，性能差は大きくなる．

この方式では，権限失効発生後の未更新データは，発生前と同じ，権限を失効されたユーザが保持している恐れのある暗号鍵で暗号化された状態である．これは未更新データの情報は権限を失効されたユーザが知っている可能性があるため漏えいしても問題ない，という考えに基づく．しかし権限を失効されたユーザが権限失効発生前に対象データを取得していない可能性もあるため，active revocation と比較するとセキュリティ面で劣るといえる．

2.3.3 両手法の比較

パフォーマンス面では lazy revocation が優れているが，セキュリティ面を考慮すると active revocation を採用すべきであると考える．そのため我々は，active revocation の安全性を維持しつつ，性能を向上させることを目指す．

3. 関連研究

encrypt-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 [7], Plutus [9], SiRiUS [10], Maat [11] がある．Plutus は lazy revocation, SiRiUS は active revocation を採用し，SNAD は両方式における性能とセキュリティのトレードオフの問題から権限失効の処理については今後の課題としている．これらのシステムは，サーバが不正を行わないという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ユーザのクライアントマシン上のみであるとしている．また，権限失効の処理はデータの所有者主導で，所有者のクライアントで実行される．我々の研究ではクライアント側の負担を軽減するために，権限失効時の再暗号化処理をストレージ側で実行するため，これらのシステムと方針が異なる．一方，Maat は Ceph ファイルシステムに対して，自動権限失効を取り入れているが，非常に短いライフタイムを前提にしておりアプローチが異る．

暗号処理専用回路を HDD (Hard Disk Drive) に組み込むことで全格納データを暗号化する技術として，Seagate の DriveTrust [12] や富士通の MTZ2CJ がある．これらは暗号化データを格納する点で encrypt-on-disk システムと同等であるが，主に HDD 紛失時におけるデータの保護が目的であるため，本論文で対象とする伝送路上のデータの機密性実現とは異なる．

4. システム要件

本研究で前提とするシステムやデータ構造の概略を述べる．

4.1 高機能分散ストレージ

我々は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 [6] 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処理能力を利用し，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．

本研究では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に encrypt-on-disk 方式を採用することを考える．クライアント側の負担の軽減のため，権限失効に伴う再暗号化処理や，暗号鍵の配布等を極力ストレージ側で行う．

4.2 プライマリバックアップ構造

耐故障化実現のため，並列ストレージシステムの各ストレージノードは，主にユーザアクセスを受けるプライマリデータと，他のノードのプライマリデータの複製であるバックアップデータをもつ．

4.3 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を行うものとする．

4.4 暗号の利用

本研究で利用した暗号法と鍵管理方法について説明する．

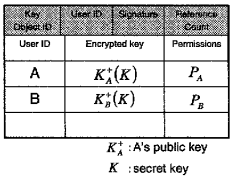


図 1 *key object*
Fig. 1 *key object*.

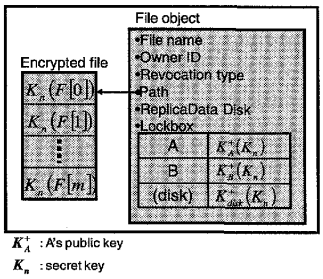


図 2 データ構造
Fig. 2 Data structure.

4.4.1 暗号法の種類と利用法

暗号は，暗号化と復号に共通の暗号鍵を用いる共通鍵暗号法と，異なる対の暗号鍵を用いる公開鍵暗号法に分類できる．共通鍵暗号は，あらかじめ暗号化側と復号側で，安全な方法を用いて鍵を共有していなければならない．一方，公開鍵と秘密鍵の対を用いる公開鍵暗号では，対のうちの片方（公開鍵）を公開することができるため，共通鍵暗号のような鍵配布の問題はない．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 [7]．

これらの特性を考慮し，一般的には公開鍵暗号を用いて共通鍵の配布を行い，その共通鍵を用いてデータのやり取りを行う場合が多い．本研究でもその方式に従う．

4.4.2 暗号鍵管理構造

暗号鍵を管理する構造としてロックボックスがある．ロックボックスは，暗号鍵を格納し，その鍵が関連するデータに対してアクセス権をも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．図 1 に SNAD [7] におけるロックボックスである *key object* の例を示す．図の 1 行目はオブジェクトのもつ属性情報を，2 行目以降がユーザごとの管理情報を示す．ここでユーザ x は公開鍵 K_x^+ と秘密鍵 K_x^- をもち，あるデータが共通鍵 K_i で暗号化されているものとし，3 行目以降の各行はユーザ ID，ユーザの公開鍵で暗号化された共通鍵 $K_x^+(K_i)$，ユーザに認可されているアクセス権の種別 P_x からなる．格納する共通鍵を獲得するには，公開鍵と対をなす，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 K_x^- を用いてのみ復号できるため，正しく認可されたユーザのみ共通鍵を獲得できる．

本論文 **6.** における実験では，この *key object* の構造を利用して共通鍵を管理する方法を用いている．一つのファイルに対し一つの *key object* が対応し，ファイルを暗号化した共通鍵を格納し，ストレージノードに置かれる．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ため，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もち，処理対象のファイルの *key object* に鍵を保持する．なお **6.** の実験では性能評価を主眼とし，ロックボックスはファイルと同じノードに，**4.5** で述べる形で格納している．この構成ではロックボックスが攻撃されると機密性は保証できないが，ロックボックスはセキュアチップや専用のノードに格納することで攻撃から守ることができる．そのようなセキュアチップや専用ノードの構成は本論文の範囲外であるが，セキュアチップや専用ノードを用いた場合の性能は，実験環境と同等またはそれ以上になるはずであり，実験は有効である．

4.5 データ構造

実現方法の一つとして実験で用いる 1 ファイル相当のデータ構造を図 2 に示す．この構造は SNAD [7] におけるデータ構造である *file object* を参考にした．ここで **4.4** で述べたとおり各ユーザ及びストレージノードは公開鍵と秘密鍵の対をもち，各ファイルは共通鍵で暗号化されている．このときファイルの暗号化粒度は全体あるいは固定長ブロック単位とする．後者の場合ファイル更新時の差分データの単位をこのブロックとすることで転送及び暗号化コストを削減できる．

ファイルへのアクセスはファイルオブジェクトを通して実行するものとする．ファイルオブジェクトにはファイル名や位置，処理方式等の情報，及び鍵を格納したロックボックスをもつ．

5. 複製データを利用した再暗号化

2.3.1 で述べた active revocation 適用時に性能が低下する問題に対し，我々は **4.** で述べたシステムを前提として構成した分散ストレージにおける，バックアップデータを利用した，効率の良い再暗号化手法 BA-Rev (Backup Assisted Revocation) [3] を提案した．本章で BA-Rev について説明する．

5.1 BA-Rev の概要

encrypt-on-disk 方式を採用した，プライマリバックアップ構造をもつ分散ストレージシステムを前提とする．なお，以降プライマリとバックアップの配置制約はないものと仮定するが，制約が存在する場合も以降の処理は可能である (RBA-Rev [3])．BA-Rev 及び既存方式の active revocation における再暗号化処理

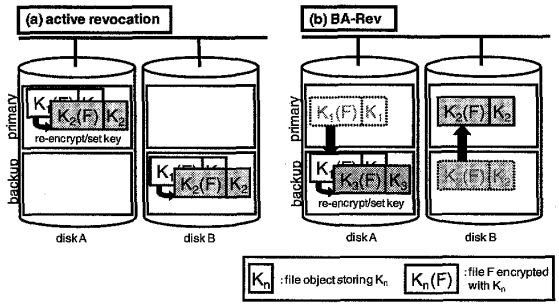


図 3　再暗号化方式の比較
Fig. 3　Comparison of revocation methods.

の流れを図 3 に示す．

BA-Rev を採用するシステムでは，バックアップデータを作成するとき，プライマリデータと同じ共通鍵で暗号化された状態ではなく，新たに生成した共通鍵であらかじめ再暗号化し，格納する．ここでバックアップデータ及びバックアップデータを暗号化した共通鍵はユーザにアクセスされず，未知であるものとする．

権限失効発生時には，対象データのバックアップデータを，バックアップデータが格納されたストレージノードのプライマリに移動して新しいプライマリデータとして設定し，もとのプライマリデータに代わってアクセスを受けるものとする．続いて，もとのプライマリデータを新しくバックアップデータとして設定するため，新たに共通鍵を生成し，再暗号化してバックアップとして格納する．

既存方式である active revocation ではプライマリデータの再暗号化中はそのファイルへのアクセスが不可能になるのに対し，BA-Rev では直ちにアクセス受付可能となる．また再暗号化処理を行うのはもとのプライマリデータが格納されていた 1 ノードのみであるため，全体での総処理コストも抑えられる．

しかし一方で，BA-Rev 環境では，active revocation 環境よりデータの更新コストが高く，性能劣化の可能性があった．これは，プライマリとバックアップの共通鍵が異なる環境では，更新がある場合，バックアップ側で差分データの再暗号化処理が必要であることが理由である．ユーザはバックアップ側で用いられる共通鍵を知らないため，差分データはプライマリデータに用いられている共通鍵で暗号化され，ストレージ側に転送される．そのため，このままバックアップデータに適用はできない．

5.2　バックアップデータへの差分データ適用の遅延

上記の問題に対するアプローチとして，バックアップデータへの差分書込みを遅延する DW(Delayed Writing) 戦略 [4]，及び差分再暗号化と書込みを遅延する DRW(Delayed Re-encrypting and Writing) 戦略 [5] を適用する．DW 戦略ではバックアップデータ用差分データを再暗号化後，DRW 戦略では転送されてきた状態のまま，メモリ上に保持し適用を遅延することで，処理を分散して性能劣化を抑える．同時に，更新頻度が高い場合，バックアップ側では複数回の更新に関する処理をまとめ適用回数を削減できる．

本論文では DW 戦略でメモリ上に保持された差分データの書込みタイミングの評価，及び DW/DRW 戦略が BA-Rev の処理に与える影響を評価する．適用が遅延された差分データに関して，いずれかのタイミングでメモリ上の差分データをファイルへ適用しなければならないが，過去の研究 [4] ではこの点を考慮しておらず，また BA-Rev の処理への影響も未評価であった．

5.2.1　差分データ適用時期の設定

メモリ上に保持した，バックアップデータへの差分データを適用する時期として，以下の 3 パターンを考える．ここで遅延された差分適用とは，DW 戦略では書込みを，DRW 戦略では再暗号化と書込みを指す．

(1)　権限失効発生時

BA-Rev では，権限失効処理時バックアップデータをプライマリデータに昇格させる．したがって，対象のバックアップデータへの差分が未適用である場合，プライマリとしてアクセスを受け付けるために権限失効発生に伴って適用する必要がある．

(2)　差分データ保存数が一定値を超えたとき

ストレージノードのメモリは有限であるため，適当なしきい値を設け，未適用の差分データの総数あるいは総サイズがしきい値を超えた場合に差分データを適用する．

(3)　特定の条件を満たしたとき

上記以外で，ある条件を満たした場合に差分データを適用する．本論文では以下の 2 項目のうちどちらかを選択し，その条件を満たしたときに適用を行う．

(3-a)　更新後一定時間経過後　　差分データ再暗号化しメモリに保存後，一定時間待機し，その後書込みを行うことで負荷を分散する．

(3-b)　負荷が一定以下　　定期的に各ストレージノードの負荷を測定し，負荷の値が一定以下のときのみ差分データの書込みを行うことで，負荷の集中を防ぐ．

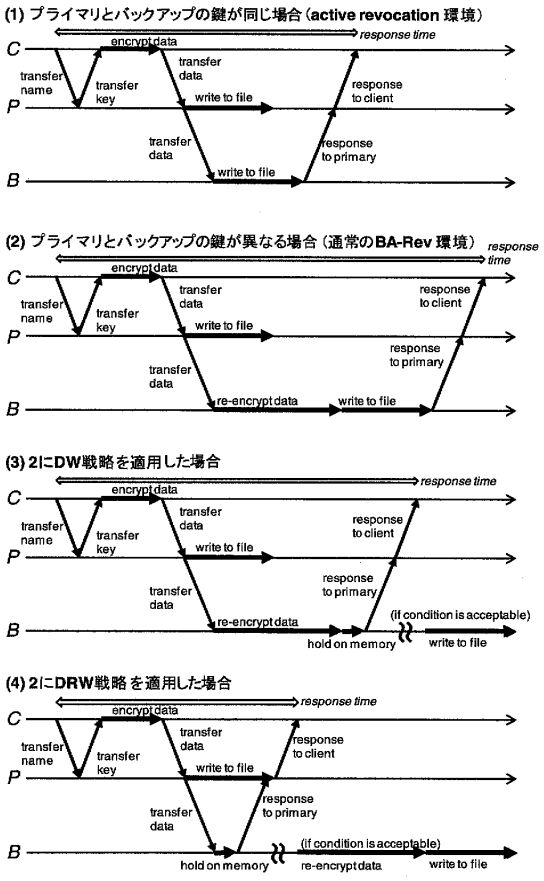


図 4　クライアント (*C*)，プライマリ (*P*) とバックアップ (*B*) における更新処理の流れ

Fig. 4　Flow of update on the client (C), primary (P) and backup (B) node.

図 4 に，(1) プライマリとバックアップで共通鍵が同じである環境（ここでは active revocation 環境），(2) 共通鍵が異なる環境（通常の BA-Rev 環境），(3) 共通鍵が異なり，かつ DW 戦略を適用した環境，(4) 同様に DRW 戦略を適用した環境における，更新発生時のクライアントマシン上，プライマリ及びバックアップでの処理の流れを示す．ここで (3) 及び (4) の環境では，上記いずれかの条件を満たした場合にメモリに保持した差分データの適用を行うことを示している．また，いずれの方式においても，ユーザからデータを受信してから，バックアップ側からの応答を受信するまでの間，排他制御によりこのプライマリデータへの他のアクセスはブロックされるものとする．

6. 実　　験

更新がある環境における BA-Rev の性能，及び DW/DRW 戦略が BA-Rev に与える影響を評価するため，実験を行った．

6.1　実験で比較する環境

以下の環境について実装，実験し，比較を行う．

i. active revocation 環境
権限失効発生時は既存方式である active revocation を実行する．あるファイルのプライマリデータとバックアップデータは同じ共通鍵で暗号化される．

ii. BA-Rev 環境
権限失効発生時は BA-Rev 方式で処理を実行する．バックアップデータはプライマリデータと異なる共通鍵で暗号化される．更新 (update) 発生時は図 4 の (2) のとおり，バックアップ側でも即時書込みを行うものとする．

iii. BA-Rev + $\mathrm{DW}_{\mathrm{raw}}/\mathrm{DRW}_{\mathrm{raw}}$
ii. に加え，更新処理は DW/DRW 戦略に従う．差分データの書き込み条件は **5.2.1** の (1)，(2) のみとする．一定間隔で未適用差分データ数を確認し，しきい値を超えていたらその分を適用する．

iv. BA-Rev + $\mathrm{DW}_{\mathrm{const}:n}/\mathrm{DRW}_{\mathrm{const}:n}$
iii. に加え，差分データ適用条件として **5.2.1** の (3-a) を用いる．制御パラメータを n とし，バックアップ側で差分データを再暗号化後，n 秒待機して適用を行う．なお，待機中に同ファイルの更新が発生した場合，待機中の差分データを後続の差分データに置き換えるものとする．

v. BA-Rev + $\mathrm{DW}_{\mathrm{load}:n}/\mathrm{DRW}_{\mathrm{load}:n}$
iii. に加え，差分データ適用条件として **5.2.1** の (3-b) を用いる．制御パラメータを n とし，一定間隔でストレージノード上のアクティブなスレッドの数を確認し，n 未満のとき未適用の差分データを適用する．この処理をスレッド数が n 以上になるまで繰り返す．

6.2　実験環境とプログラム

表 1 のノードからなる PC クラスタ上で動作する，encrypt-on-disk 方式のファイルサーバ及びクライアントプログラムを作成した．実験に用いた各パラメータを表 2 に示す．

サーバプログラムは **6.1** の環境を構築する．データの初期配置は chained declustering [13] に従い，バッ

表 1 ストレージノード諸元
Table 1 Storage node data.

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

表 2 固定パラメータ
Table 2 Fixed parameter.

公開鍵	RSA 1024 bit
共通鍵	AES 128 bit
暗号化モード	ECB
パディング	PKCS5Padding
ストレージノード数	3
ノード当たりデータサイズ	1 MB×500
差分データサイズ	100 KB
Zipf 母数 θ	0.7
未適用差分データ数/スレッド数確認間隔	10 sec

クアップデータは，プライマリデータを格納するノードの論理的に“右隣”となるノードに配置するものとする．ただし処理中の配置制約はなく，BA-Rev の処理で配置が変わる可能性はある．

クライアントプログラムはサーバプログラムに対し，ファイル獲得 (get)，ファイル更新 (update)，権限失効，及びファイルの格納とユーザの認可を要求できる．また get と update は応答時間を計測できる．ここで応答時間はクライアント側で以下の処理に掛かる時間と定義する．なお update に関しては図 4 も参照されたい．get は update と異なり各環境で処理は同じであるため，比較図は省略する．

（1） **update**

1. サーバに対象ファイル名を送信する．
2. サーバから対象ファイルのプライマリデータに使われている，クライアントの公開鍵で暗号化された共通鍵を受信する．
3. 暗号化された共通鍵を秘密鍵で復号する．
4. 差分データを作成し，共通鍵で暗号化してサーバに送信する．
5. サーバ側の終了通知を受信する（終了通知のタイミングは図 4 参照）．

（2） **get**

1. サーバに対象ファイル名を送信する．
2. サーバから暗号化ファイル（プライマリデータ）及びクライアントの公開鍵で暗号化された共通鍵を受信する．
3. 暗号化された共通鍵を秘密鍵で復号する．
4. 共通鍵で暗号化ファイルを復号する．
5. ファイルを記憶装置に書き込む．

get，update の実行間隔は，平均到着間隔 $1/\lambda$ の指数分布 $f(t) = \lambda e^{-\lambda t}$ に従い決定する．ここで固定の get:update 比に従いアクセスを実行する．また対象ファイルは，偏りをもたせるために，ノードに格納されたファイルからパラメータ θ に従う Zipf 分布 [14] に基づき選択される．

共通鍵暗号アルゴリズムは既知平文攻撃等の攻撃に強いとされる AES を選択した．また，**4.5** で挙げたように，更新処理を固定長ブロック単位で行うために，暗号化モードとして共通鍵アルゴリズム固有のブロックサイズごとに独立して暗号化を行う ECB を用い，このブロックを更新処理の単位として用いた．

6.3 実験 1：通常時のアクセスの応答時間

各環境における通常時の応答性能を評価するための実験を行った．

6.3.1 実験方法

3 台のストレージノードに，各ノードのプライマリデータ数が 500 になるよう 1 MByte のファイルを格納する．この状態で，1 ストレージノードに対し一つのクライアントノードから，一定の get:update 比 (50:50) に従い，アクセスを行う．アクセスの平均到着間隔 $1/\lambda$ を 500 ミリ秒から 150 ミリ秒まで変化させて，平均応答時間の変化を観測した．

6.3.2 考　　察

DW 戦略または DRW 戦略を適用した環境での update の平均応答時間を図 5，図 6 に示す．比較のため，active revocation 及び BA-Rev 環境での平均応答時間は両図に記載した．また get の平均応答時間を同様に図 7，図 8 に示す．

active revocation 環境と BA-Rev 環境における update の平均応答時間を比較すると，BA-Rev 環境の応答時間が大きい．プライマリとバックアップに用いる共通鍵が同じである active revocation 環境に対し，共通鍵が異なる BA-Rev 環境ではバックアップ側で差分データの再暗号化処理が必要であり，この処理に時間が掛かるためである．

通常の BA-Rev と DW 戦略を適用した各環境で update の平均応答時間を比較すると，低負荷時ではわずかに DW 戦略を適用した環境の方が性能が良かっ

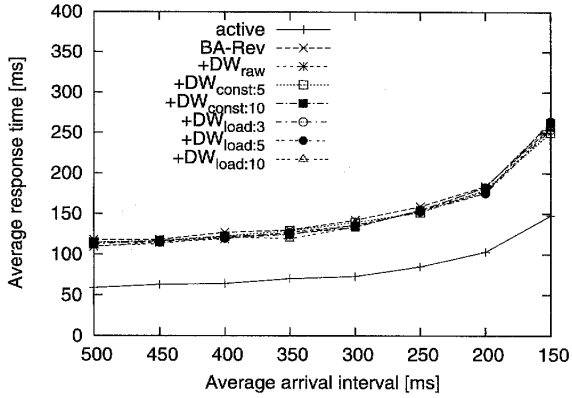


図 5　update の平均応答時間（DW 戦略適用）
Fig. 5　Average response time of update with DW.

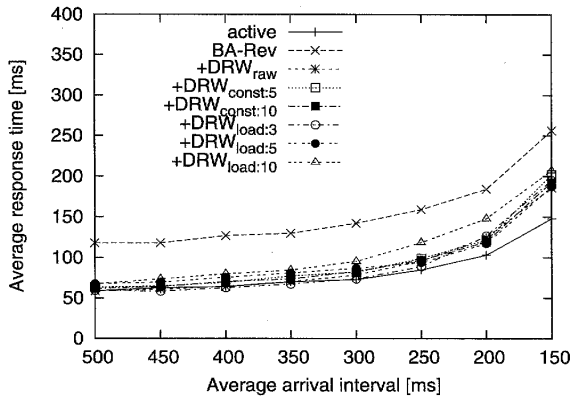


図 6　update の平均応答時間（DRW 戦略適用）
Fig. 6　Average response time of update with DRW.

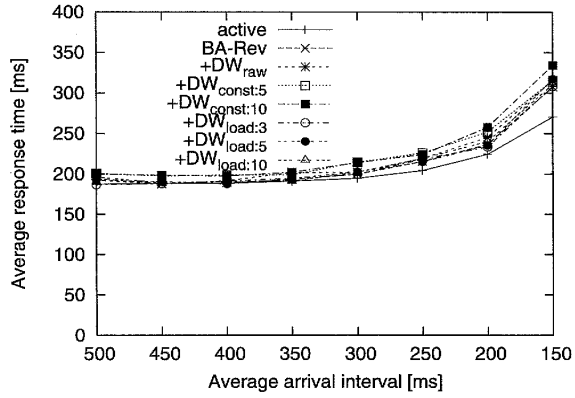


図 7　get の平均応答時間（DW 戦略適用）
Fig. 7　Average response time of get with DW.

たものの，大きな差は出なかった．これは，本実験の実装では，あるファイルで複数回更新が発生した場合，キャッシュの利用により記憶装置との同期がとられず，高速に処理されているためと考える．

一方 DRW 戦略を適用した環境における update の平均応答時間は，BA-Rev と比較して大きく改善した．これは時間が掛かっていた再暗号化処理を遅延したことによる効果である．また適用条件による差異は，$DRW_{load:10}$ では負荷が上昇するとまとめて差分データを適用してしまう場合があるためわずかに性能が落ちるものの，他の環境では適用条件ごとに一長一短があり，大きな差が現れない．

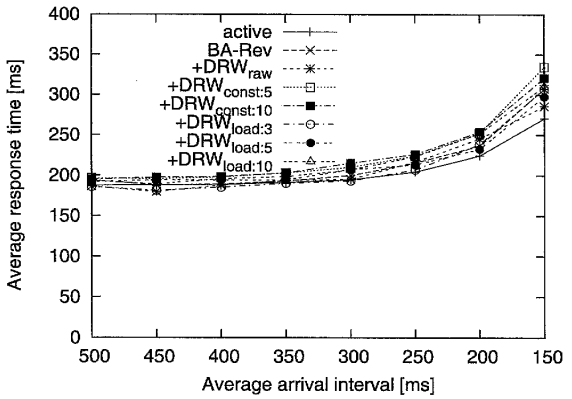


図 8　get の平均応答時間（DRW 戦略適用）
Fig. 8　Average response time of get with DRW.

get の処理に関しては，いずれの環境でも処理が同じであるため，高負荷時にはバックアップでの update に関する処理の影響でわずかに差が現れるものの，全体として大きな差は見られない．

この実験により，BA-Rev 環境での更新の応答時間に，暗号処理の性能が大きく影響を与えていることが分かる．例えば，低負荷時の update 応答時間のうち，80 パーセント強がクライアント上及びバックアップ側における暗号処理に掛かっていた．暗号処理時間は CPU 性能に依存する．そのため，ストレージノードの性能を考慮することでも更新性能を改善できる．表 1 のストレージノードの環境と，高性能サーバ機 (CPU: AMD Opteron 248 (2.2 GHz) × 2) で，100 kByte のデータの暗号化及び再暗号化処理の実行時間を比較したところ，後者は前者と比較して処理時間が約 40 パーセント改善した．

active revocation への DW 戦略適用も可能だが，active revocation の場合権限失効処理の性能改善はない．なぜなら，差分データの複製側に対する書込みを遅延しても，権限失効処理発生時にはプライマリ及びバックアップで通常と同様の再暗号化処理を必要とするからである．一方，update 処理については，DW 戦略を適用することで，複製側の書込み処理を待つ必要がないため，応答速度向上が期待できる．しかし，プライマリデータとバックアップデータで一時的に不整合が生じる上に，性能向上を期待できるのは複製側

が高負荷な場合だけなど限定的であるため，実験は省略する．また，active revocation における update 処理においては，バックアップ側での差分データの再暗号化が発生しないため，DRW 戦略は適用できない．

6.4 実験 2：権限失効を集中して発生させた場合の比較

一つのストレージノードで権限失効が発生した場合における，各環境での性能比較のため実験を行った．

6.4.1 実験方法

6.3.1 と同様に 3 ストレージノードにファイルを格納し，3 クライアントからアクセスを行う．ここでは平均到着間隔は 400 ミリ秒，DW 戦略における制御パラメータ n は 5 に固定する．また初期状態ではノード A，B，C のプライマリに対するバックアップはそれぞれ B，C，A にある．

アクセスを実行している状況で，アクセスを実行するクライアントとは異なるクライアントで一つのストレージノード（ここではノード B）中の，ランダムに選択した 50 個のファイルに関して権限失効を実行し，get と update の応答時間の変化を測定した．ただし権限を失効されるユーザはこの 4 クライアントとかかわりのない仮想的なユーザであるとする．

6.4.2 考　　察

上記の実験を 6 回行った．get 及び update それぞれについて，各ノードごとの，権限失効発生から収束までの期間を含んだ 100 アクセス分の応答時間の平均，及び各実験の平均の 95 パーセント信頼区間を図 9，図 10 に示す．

権限失効対象ファイルをもつノード B 及びそのバックアップデータがあるノード C に関して，active revocation と比較して提案方式の各環境で，get，update 共に性能が改善された．これは BA-Rev ではノード C で再暗号化処理が不必要である点，ノード B では再暗号化処理が必要であるものの，対象がアクセスされないバックアップデータである点が理由として挙げられる．

ノード A に関しては，active revocation と比較して提案の各環境の性能は同等，あるいは若干劣化した．これは **6.4** で示した，通常時の性能差に起因するものと考える．ただしノード B，C と比較すると，手法間の差は小さい．

BA-Rev と DW/DRW 戦略を適用した環境を比較すると，get では特にノード B で，DW/DRW 戦略がより良い結果を示した．これはノード B では権限失効

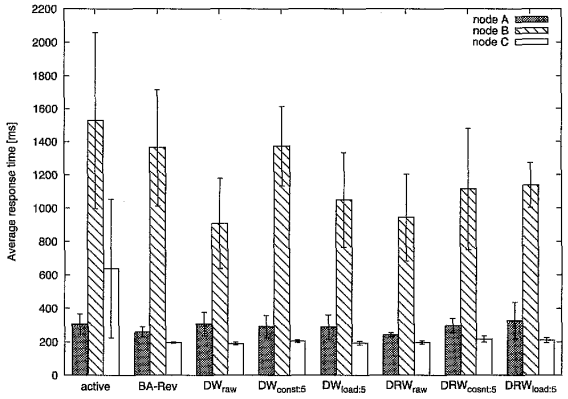


図 9　権限失効が集中した場合の平均応答時間 (get)
Fig. 9　Average response time of get under concentrated revocation.

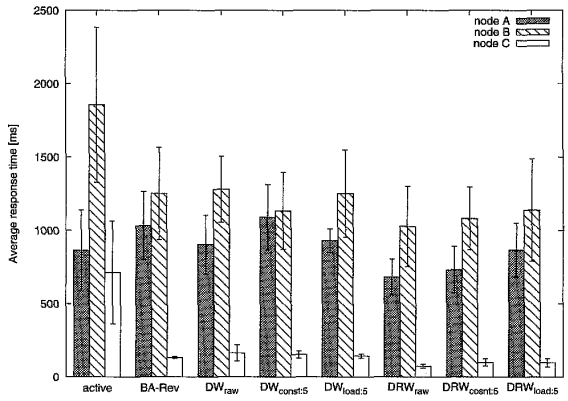


図 10　権限失効が集中した場合の平均応答時間 (update)
Fig. 10　Average response time of update under concentrated revocation.

発生に伴い多量の再暗号化及びディスク I/O が発生するのに対し，両戦略では適用遅延により差分適用回数を削減でき，更なる性能劣化を引き起こしにくいためだと考える．一方 update に関しては，差分データ再暗号化まで遅延する DRW 戦略の環境が，BA-Rev 及び DW 戦略の環境より性能が良い．これは update の処理が再暗号化処理による影響を受けやすく，差分再暗号化遅延の影響が大きいためだと考える．

なお本論文では，DW/DRW 戦略における差分適用条件の差異による性能評価は割愛する．本実験及び次節の実験における平均及び信頼区間を考慮すると，条件間に明確な優劣を見出せなかった．そのため，実験方法や回数，条件を改めて評価を行うことを今後の課題とする．

6.5 権限失効が分散して発生させた場合の比較

複数のストレージノードにわたって同時に権限失効が発生する場合を想定し，実験を行った．

6.5.1 実験方法

実験2と同様にデータを格納し，アクセスを行っている環境で，3ストレージノードすべてで同時に，それぞれランダムに選択した15ファイルについて権限失効を発生させ，応答時間の変化を観測した．

6.5.2 考　　察

この実験を6回実行した．get及びupdateそれぞれについて，権限失効発生から収束までの期間を含んだ100アクセス分の，全ノード応答時間の平均，及び各実験の平均応答時間の95パーセント信頼区間を図11，図12に示す．

6.4 と同様，active revocationと比較して，BA-Revを用いた各環境ではget，updateともに性能がよい．これはバックアップ側で再暗号化が不必要である点，プライマリ側の再暗号化がバックグラウンドで処理される点の両方が理由であると考える．BA-RevとDW/DRW戦略適用環境の比較では，updateに関してDRW戦略が他環境より良い結果となった部分を除き，全体に大きな差が出ない．これは，この実験では権限失効要求を全ノードに分散させた分，1ノード当りの負荷が小さいためであると考える．また，この結果より，1ノード当りの権限失効対象ファイル数が少なければ，BA-Rev及びいずれのDW/DRW戦略適用環境でも，条件によらず安定してactive revocationより良い性能を実現できることを表しているといえる．

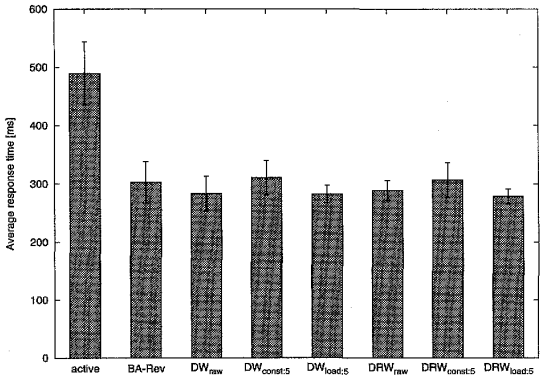


図11　権限失効が分散した場合の平均応答時間 (get)
Fig. 11　Average response time of get under distributed revocation.

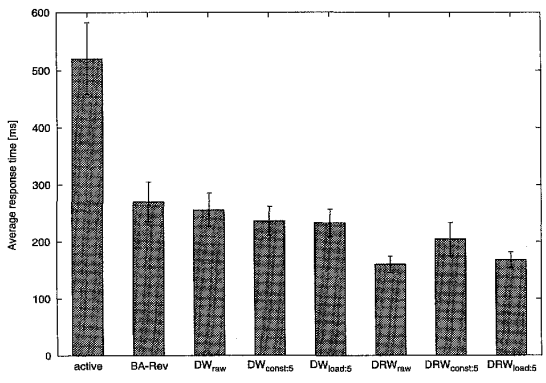


図12　権限失効が分散した場合の平均応答時間 (update)
Fig. 12　Average response time of update under distributed revocation.

7. 信頼性の評価

DW，DRW戦略では，プライマリとバックアップにおけるディスク上のデータに，一時的に不整合が生じる．そのため，ディスク故障と電源供給系の故障の同時発生による，データ喪失の可能性がある．例えば，通常一つのディスクが故障した場合，バックアップ側の未適用差分データをバックアップデータに適用し，そのバックアップデータを用いて正常復帰が可能である．しかし，電源系の故障によってメモリ上の未適用差分データが失われた状態でディスク故障が発生した場合や，ディスク故障後に未適用差分データを適用し終わる前に電源系が故障した場合は，故障ディスクに対するバックアップには差分適用前の古いデータしか残っておらず，最新のデータが失われる．

これらの信頼性評価のため，DW，DRW，及び両戦略を適用せず複製側の差分適用遅延を行わないactive revocation及びBA-Revの環境 (CD：Chained Declustering) におけるMTTDL（Mean Time To Data Loss：平均データ喪失時間）を見積り，比較した．算出方法及び詳細は[15], [16]及び[17]を参照されたい．図13に，ノード数 (N) を変化させた場合の，各環境におけるMTTDLを示す．$\mathrm{MTTDL_{Disk}}$，$\mathrm{MTTDL_{Power}}$ はそれぞれディスク故障のみ及び電源系故障に起因するMTTDLを表し，DW及びDRWではこれらを独立に算出した．

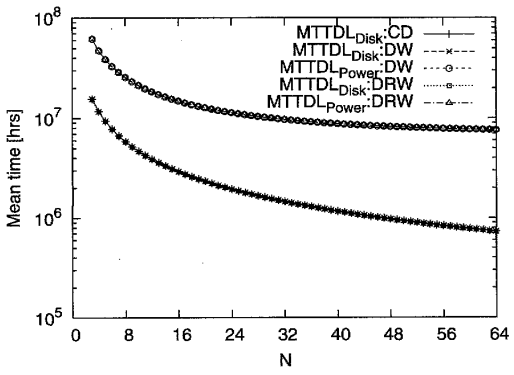


図13　各環境におけるMTTDL
Fig. 13　MTTDL on each environment.

グラフでは各環境 (CD, DW, DRW) における $MTTDL_{Disk}$ の値がほぼ重なっており，これらの差は1パーセント以下と非常に小さい．また，DW及びDRWにおける電源故障によるデータ喪失を示す $MTTDL_{Power}$ を見ると，$MTTDL_{Disk}$ よりもはるかに大きい．例えば，今回の見積り条件で $N = 32$ の場合，図13で各環境の $MTTDL_{Disk}$ がおよそ 10^6 であるのに対し，$MTTDL_{Power}$ はおよそ 10^7 である．このため，システム全体の信頼性評価おける電源故障による影響はディスク故障に比べるとはるかに小さく影響をほぼ無視できることが分かる．

以上より，DW戦略及びDRW戦略で差分データの適用を遅延し，メモリ上に保持することによる信頼性への影響は，非常に小さいといえる．

8. むすび

本論文では，encrypt-on-disk方式を採用する分散ストレージシステムにおける，高速かつ効率の良い権限失効時再暗号化処理方式BA-Revに，更新時にバックアップ側の差分データの書込みを遅延するDW戦略，及び差分データの再暗号化と書込みを遅延するDRW戦略を適用し，更新がある環境での評価を行った．BA-Revではバックアップデータをあらかじめ新しい共通鍵で暗号化しておくことで，権限失効対象ファイルへアクセスできない状況を即時回復でき，同時に再暗号化処理を行うノード数が減少するため，複数ノードでの性能低下を抑える．PCクラスタを用いた実験により，更新がある環境でもその効果を確認できた．また，特にDRW戦略を適用し，再暗号化処理も含めてバックアップ側の差分データ処理を遅延し，複数の更新に対する処理を集約することで，active revocationの環境と同等の更新性能を維持しつつ，更に権限失効時にBA-Revの性能を向上できることを確認した．

また，DW/DRW戦略適用環境と既存方式の環境とのMTTDLを比較し，各環境での信頼性に大きな差がないことを示した．つまり，提案方式ではバックアップ側で未適用差分データをメモリ上に保持するため，電源系とディスクの故障によりデータを喪失する可能性があるが，信頼性評価におけるその影響は非常に小さいことが示された．

以上より，提案手法がネットワークストレージにおいて，信頼性とセキュリティを保ちながら高い性能を提供できることが確認された．今後，サイズやタイプの異るファイルを多数蓄積し，様々なアプリケーションから多様なアクセスが来るようなより実際に近い環境での評価が必要となる．

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，及び文部科学省科学研究費補助金特定領域研究 (19024028) の助成により行われた．

文　献

[1] E. Riedel, M. Kallahalla, and R. Swaminathan, "A framework for evaluating storage system security," FAST '02: Proc. 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 Research, ArXiv Computer Science e-prints, 2004.

[3] 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，Feb./March 2007.

[4] K. Takayama, D. Kobayashi, and H. Yokota. "Consideration of experimental evaluation about encrypted replica update process," International Workshop on Advanced Storage System 2007 in Conjunction of Proc. Second IEEE International Conference of Digital Information Management, pp.545–550, IEEE, Oct. 2007.

[5] K. Takayama and H. Yokota, "Performance and reliability of a revocation method utilizing encrypted backup data," Proc. PRDC'09, pp.151–158, 2009.

[6] H. Yokota, "Autonomous disks for advanced database applications," Proc. International Symposium on Database Applications in Non-Traditional Environments (DANTE'99), pp.435–442, Nov. 1999.

[7] E. Miller, D. Long, W. Freeman, and B. Reed, "Strong security for network-attached storage," FAST '02: Proc. 1st USENIX Conference on File and Storage Technologies, pp.1–13, Berkeley, CA, USA, 2002.

[8] K. Fu, Group sharing and random access in cryptographic storage file system, Master's thesis, MIT, 1999.

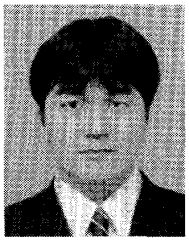
[9] M. Kallahalla, E. Riedel, R. Swaminathan, Q. Wang, and K. Fu, "Plutus: Scalable secure file sharing on untrusted storage," FAST '03: Proc. 1st USENIX Conference on File and Storage Technologies, pp.29–42, 2003.

[10] E.-J. Goh, H. Shacham, N. Modadugu, and D. Boneh, "SiRiUS: Securing remote untrusted storage," Proc. Internet Society (ISOC) Network and Distributed Systems Security (NDSS) Symposium, 2003.

[11] A.W. Leung, E.L. Miller, and S. Jones, "Scalable security for petascale parallel file systems," Proc.

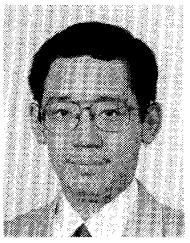
SC07, pp.1–12, 2007.
[12] Seagate. Drivetrust^{tm} technology: A technical overview. http://www.seagate.com/docs/pdf/whitepaper/TP564_DriveTrust_Oct06.pdf
[13] H.-I. Hsiao and D.J. DeWitt., "Chained declustering: A new availability strategy for multiprocessor database machines," Proc. Sixth International Conference on Data Engineering, pp.456–465, Washington, DC, USA, 1990.
[14] D.E. Knuth, Sorting and Searching, Addison-Wesley Publishing Company, 1973.
[15] E.M. Macdonald, Designing Reliable Large-Scale Storage Arrays, Master's thesis, Florida State University, College of Engineering, 2007.
[16] F. Bodi, ""DC-Grade" reliability for UPS in telecommunications data centers," Telecommunications Energy Cnference 2007. INTELEC 2007. 29th International, pp.595–602, 2007.
[17] 高山一樹，暗号化データ格納分散ストレージにおける性能とセキュリティの両立に関する研究，Master's thesis，東京工業大学大学院，Jan. 2009.

（平成 21 年 6 月 8 日受付，10 月 5 日再受付）



高山　一樹

平 19 東工大・工・情報工学卒．平 21 同大大学院・情報理工・計算工・修士課程了．現在（株）エヌ・ティ・ティ・ドコモ．セキュアストレージ，暗号化データの権限失効処理に関する研究に従事．



横田　治夫　（正員：フェロー）

昭 55 東工大・工・電物卒．昭 57 同大大学院・情報・修士課程了．同年富士通（株）．同年 6 月（財）新世代コンピュータ技術開発機構研究所 (ICOT)．昭 61（株）富士通研究所．平 4 北陸先端大・情報・助教授．平 10 東工大・大学院情報理工・助教授．平 13 東工大・学術国際情報センター・教授，現在に至る．工博．主として分散インデキシング，データ工学向けアーキテクチャ，高機能ストレージシステム，ディペンダブルシステム等に関する研究に従事．元本会データ工学研究専門委員会委員長．ACM SIGMOD 日本支部長．日本データベース学会理事．情報処理学会フェロー．人工知能学会，IEEE，ACM 各会員．