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

2007－DPS－133 (11)
2007／11／27

# 仮想大規模ストレージにおけるセキュリティ

上原 稔
東洋大学工学部情報工学科

企業は内部統制のためシンクライアントの導入を進めている。シンクライアント環境では、大容量ストレージが必要である。しかし、従来の大容量ストレージは高価である。そこで、我々は PC データグリッドに基づくストレージを提案し、これを VLSD(Very Large-Scale Disk)と名づけた。我々は 70TB の VLSD を試作した。その経験から、VLSD にはセキュリティ上の問題が存在することが判明した。例えば、従来の VLSD では、なりすまし、盗聴、改ざんが可能であった。本論文では、このようなセキュリティ上の問題を解決するために、VLSD のためのセキュリティフレームワークを提案する。このフレームワークを用いることで、これらのセキュリティ問題を解決することができる。また、その性能を評価した。

# Security in Virtual Large-Scale Disk

Minoru Uehara
Dept. of Information and Computer Sciences, Toyo University, Japan

Many enterprises proceed to introduce thin clients for internal control. In a thin client environment, a large capable storage is required. However, such storage is expensive. So, we propose novel storage based on PC data grid, and then we call it VLSD (Very Large-Scale Disk). We developed prototype storage of 70TB using VLSD. In our experience, we found that there are several security issues in VLSD. For example, conventional VLSD allows becoming, wiretap and alteration. In this paper, we propose a security framework for VLSD in order to solve these issues on security. Using, this framework, these issues can be solved. In addition, we evaluate the performance of this security framework.

## 1. はじめに

近年、内部統制のためにシンクライアントを導入する企業が増えてきた。シンクライアント環境では、サーバがストレージを一元的に管理する。そのため、クライアント数に比例してサーバのストレージ容量は増加する。一般的にユーザは実現方法を関知しない。ゆえに、シンクライアントにもファットクライアントの機能を要求する。近年、HDD 技術は急速に進歩し、ノート PC にも最低 60GB の HDD が搭載されている。そこで、もし 1000 人のユーザが 60GB の容量を要求したら、サーバのストレージ容量は合計で 60TB になる。

普及品には、このような大容量ストレージは存在しない。一般的に、このような大容量ストレージを構築するには、NAS や SAN 等を用いる。しかし、これらのストレージは高価である。例えば、HDD ユニットの単価は同一容量の普及品の 10 倍を超える。その結果、60TB の SAN を構築するには 2 億円以上かかる。

このようにストレージの経費は極めて高い。前の例の場合、実際には、1 人あたりの容量を制限することでサーバの容量を削減する。よって、シンクライアントではファットクライアントと同等の生産性は得られない。

また、シンクライアントは、しばしば HDD を除いた PC であることが多い。普及品の HDD は非常に安いため、このようなシンクライアントに価格優位性はない。

そこで、本論文では、通常の PC（つまりファットクライアント）を用いてシンクライアント環境を構築する方式を提案する。この方式では、クライアントは HDD を装備するファットクライアントである。しかし、その HDD はそのクライアントのために使用されるわけではない。ファイルサーバはクライアントが内蔵する HDD を統合し、1 つの大規模ストレージを構成する。これにより高価な SAN は不要となる。我々は、このようなストレージ構成を VLSD(Virtual Large-Scale Disk)と呼ぶ。また、そのための必要な機能を備えたツールキットを Java で開発した。VLSD ツールキットは 100% pure Java で記述され、プラットフォームに依存しない。Windows と Linux のいずれでも動作する。多様なクラスを組み合わせることで 8EB までの大容量ストレージを実現する。我々は、VLSD ツールキットを用いて 500 台の PC からなる PC 教室のために 70TB のストレージを試作した。

したがって、VLSD は、60GB の HDD を備えた 1000 台の PC を用いて 60TB サーバを構成できる。しかし、実際の運用では、内部統制を実現するため、セキュリティが問題となる。従来の VLSD には、セキュリティ機能が不足していた。そこで、本論文では、VLSD にセキュリティのための基本的な機能を追加し、安全な運用を可能とする。この結果、VLSD のみで安全な大規模ストレージを構築することが可能となった。しかし、性能を考慮すると実際の運用では VLSD と既存技術を組み合わせて運用することが望ましい。

本文の構成は以下の通りである。2 章では関連研究を紹介する。3 章では VLSD について述べる。4 章では VLSD のセキュリティについて述べる。5 章ではセキュリティ機能を評価する。最後に結論を述べる。

## 2. 関連研究

大規模ストレージの研究は古くから行われている。安価な複数の HDD を用いて 1 つの大容量ストレージを構築する方式の 1 つに RAID[3][4]がある。大規模ストレージを構築するには耐故障性を実現する必要がある。RAID は耐故障性と高速性を実現する。RAID は、それが Patterson らにより提案されたときには高速性より経済性が評価された[3]。しかし、今日、RAID は経済性より高速性を重視している。普及品の RAID では、接続可能な HDD 台数が 4-6 台しかない。したがって、我々が目標とする大規模ストレージを RAID で実現することは難しい。

RAID の規模を増加させる技法に階層型 RAID がある。階層型 RAID には AutoRAID[5]、HiRAID[6]などがある。HiRAID は AutoRAID より汎用である。本文では HiRAID に基づく階層型 RAID を採用する。階層型 RAID では、通常の RAID より信頼性を向上させることが可能となる。例えば、通常の RAID5 は 1 耐故障だが、RAD55（2 階層の RAID5）は 3 耐故障である。RAID55 の信頼性は RAID6 を上回る。一般に n 階層 RAID5 の耐故障性は $2^n$-1 である。しかし、RAID55 の容量効率は RAID6 より低い。

階層型 RAID では、階層を増やすほど信頼性が高まり、同時に規模も拡大する。例えば、n 層の RAID があり、各層が m 個の RAID5 からなるとすると、その規模は $m^n$であり、信頼性は $2^n$-1 である。しかし、階層が増えると、容量効率は低下し、経済性が急速に劣化する。RAID の経済性は容量効率で決定される。m が十分大きいとき、その容量効率は以下の式で表される。

$$\left(\frac{m-1}{m}\right)^n = \left(1-\frac{1}{m}\right)^n \approx 1-\frac{n}{m}$$

ゆえに、m>>n であることが望ましい。本研究では、複製方式との分岐点である容量効率 50%以上を目標とする。しかし、普及品 RAID では、m が小さく規模の拡大が困難である。

大規模な RAID が高価な原因は専用ハードウェアにある。本研究では、ハードウェア RAID に代わりソフトウェア RAID を採用する。一般にソフトウェア RAID の性能はハードウェア RAID より低い。しかし、RAID コントローラに高性能プロセッサを用いることでソフトウェア RAID の性能を改善することができる。我々は汎用 CPU に基づくソフトウェア RAID を採用する。

分散ディスクシステムには Petal、OBSD などがある。

Petal[7]は、仮想的なコンテナをサポートするブロックレベルの分散ストレージである。14 台の 4.3GB HDD を用いた試作品は 155Mbps ATM 上で 43.1MB/s の性能を達成した。Petal は RAID01E に類似の chained-declustered data access をサポートしている。Petal はディスクを透過的にアクセスするため global map(GMap)を用いる。GMap には scalability の点で問題がある。VLSD の実装は Petal と異なるが、VLSD は Petal の動作を模倣できる。

OBSD(Object Based Storage Devices)[8]は仮想デバイスの一種であり、可変容量の NAS として利用される。OBSD では、ファイルは複数のオブジェクトに分解され、デバイス間に分散される。文献[8]では、数千ノードからなる 2PB ストレージを想定し、そのために必要な技法として FMC(Fast Mirroring Copy)と LPB(Lazy Parity Backup)を提案している。

通常のディスクは $10^{14}$～$10^{15}$アクセスごとに 1 回ビットエラーを発生する。ビットエラーはディスク全体の故障ではない。故障ディスクが存在しなくても 2BD では 1 時間に 1 回エラーが起きる。また、2PB では、単純な RAID はリビルド時間が長くなるため適用できない。例えば、5MB/s の I/F で 500GB を修復すると 1 日以上($10^5$[s]=1.16[day])かかる。これでは、縮退モードでの動作期間「無防備な窓(window of vulnerability)」が大きすぎる。ディスクの MTTF を $10^5$[h]と仮定すると 1 日の間に 2 台目のディスクが故障する確率は 0.1%である。また、MTTDL(Mean Time To Data Loss)は 3 年以下になる。

OBSD では、上位ファイルシステムで冗長性を持たせる。オブジェクトごとに 2～3 の複製を維持する。通常の複製（例えば、RAID1）では、修復のために 500GB のディスク全体をコピーするため、5MB/s の I/F で 1 日かかる。一方、FMC では、オブジェクト単位で並列にコピーする。例えば、500MB のオブジェクト単位でコピーする場合、同 I/F でも 100 秒で終了する。ただし、このためには、複製されるデータがディスクの一部である（利用率が低い）こと、多数の（小さな）ディスクが存在することが条件となる。OBSD は仮想ディスクであるため物理ディスク全体を複製しない。使用中のデータまたは必要なデータのみコピーすることができる。また、N 台の物理ディスクに OBSD が分散していれば見かけの I/F 転送速度を N 倍に上げることができる。ただし、ネットワーク層などシステムのいずれかにボトルネックがないことが条件となる。仮に 500GB のディスク 4000 台を用いて 2PB を構成し、1 台のディスクに 1GB の OBSD が 500 個対応していたとすれば、500 並列(2.5GB/s=20Gbps)でデータをコピーする必要がある。このような技術は 2007 年現在まだ普及していない。

OBSD では、さらにオブジェクト間でパリティをとることで信頼性を高めている。パリティは LPB で効率的に更新する。

文献[8]では、FMC、LPB などにより 2 way mirroring でも十分実用可能であるとの結論を得ている。しかし、ミラーでは容量効率は 50%に満たない。文献[8]では、2002 年時の容量コストを$100/100GB、2005 年時のコストを$10/100GB と仮定しているが、

実際には2007年現在$100/300GBである。つまり、容量コストは予想ほど低下していない。よって、安易に複製を多用することは経済的でない。

近年、大規模ストレージの研究がデータグリッドの分野で行われている。

Gfarm[9]はグリッド上で大規模ストレージ（データファーム）を実現する広域仮想ファイルシステムである。ここで、仮想ファイルシステムとは下位層の物理的ファイルシステムに対して論理的なファイルシステムを仮想的に提供するサービスをさす。複製を活用することで耐故障性と並列分散処理性能を向上させている。すべての資源を複製すればHDDの利用効率は50%以下となる。

Lustre[10]はオープンソースのクラスタファイルシステムである。Lustreファイルシステムは MDS(Meta Data Server)と OST(Object Storage Target)から構成される。メタデータはデータの場所を示す。メタデータをアクセスするオーバーヘッドが発生するが、透過性を実現できる。Lustre は NFS より大規模な分散ファイルシステムを構築できる。東工大の TSUBAME では Lustre を用いて 1PB のストレージを実現している[11]。

グリッドの多くは UNIX しかサポートしていない。Windows クライアントを直接接続することは困難である。通常、Windows クライアントからアクセスするには Samba を経由する。

大規模ストレージを構築するには、大容量をサポートしたファイルシステムが必要である。64bit 以上のファイルシステムには Windows の NTFS、Linux の XFS、Solaris の ZFS などがある。本研究では、NBD を標準サポートするためサーバ OS として Linux を採用したため、XFS を採用した。複数の OS が混在する環境では、Windows、UNIX に対してそれぞれ CIFS、NFS を用いてファイル共有する。

OS のシステムコールをディスクレベルでフックするには、NBD(Network Block Device)が適している。NBD は Linux で普及している仮想ディスクである。NBD は Windows にも移植されている。NBD はサーバとクライアントからなる。NBD サーバはユーザプロセスであり、スーパーユーザでなくても実行できる。任意のデバイスまたはファイルを仮想的なブロックデバイスとしてネットワーク上に公開する。一方、NBD クライアントはカーネルに依存するため、実行するにはスーパーユーザの権限を必要とする。NBD クライアントは v-node を用いで NBD デバイスへの読み書きを NBD サーバへ伝達する。NBD デバイスはメジャー番号 43 で予約されている。NBD デバイスの作成にはスーパーユーザの権限が必要である。

## 3. VLSD ツールキット

VLSD はディスクレベルで大規模ストレージを構築するための Java ツールキットである。多様なクラスを組み合わせることで 8EB までの大容量ストレージを実現する。

VLSD の第 1 版は十分な性能が得られなかった。そこで、インターフェースを刷新し第 2 版を作成した[1][2]。これら文献にはそれぞれ視点の異なる評価が掲載されている。本論文では、第 2 版をベースにセキュリティ機能を追加する。本節では、ベースとなる第 2 版について説明する。

VLSD には以下のクラス/インターフェースが存在する。

Disk: すべての仮想ディスクに共通のインターフェースを定義する。

FileDisk: 単一ファイルからなる固定容量ディスクである。論理容量は物理容量に等しい。ただし、ディスクの最大容量を動的に増やすことはできる。

VairableDisk: 1 つの単一ディスクを用いて可変容量ディスクを構成する。ディスクは先頭からオンデマンドに使用される。VariableDisk は k-tree 構造を持つ。各ノードサイズは 8KB である。それゆえ、葉ノードは 8KB のデータを保存する。中間ノードは 1024 の 64bit ポインタを保存する。各ノードは必要に応じて動的に割り当てられる。木の最大深さは 6 である。理論上、最大容量は 8EB である。しかし、中間ノードのオーバーヘッドが 0.1%であるため、最大容量は 8EB より小さい。

DiskCache: メモリを用いたディスクのキャッシュ。LRU に基づく。

RAID0,1,4,5,6: 代表的な RAID クラス。RAID0 には耐故障性がない。RAID1,4,5 は 1 耐故障である。RAID6 は 2 耐故障である。停止故障をマスクする。

VotedRAID1: RAID1 に似ているが、任意故障をマスクするために多数決を行うことが異なる。そのため、3 台以上のディスクを必要とする。RAID 中、最も信頼性が高い。

JBOD: 複数のディスクを連結した 1 つのディスクを作成する。冗長性がない点で RAID0 に似ているが、ストライピングしない点で RAID0 と異なる。ストライピングしないため、異なる容量のディスクを連結できる。性能は RAID0 より劣る。

SynchronizedDisk: ディスクを排他制御するラッパーである。

MemoryDisk: メモリを用いた小さなディスクである。

NullDisk: 内容が不変であり、0 で埋められた仮想ディスク。物理容量は 0 である。

NBDDisk/NBDServer: NBDServer は NBD サーバであり、NBDDisk はそのクライアントである。これらは互いに NBD プロトコルで通信する。NBD プロトコルはコネクション指向のプロトコルであるため、1 つのコネクションに 1 つのポートを必要とする。それゆえ、ポート数がボトルネックとなる可能性がある。

RemoteDisk/DiskServer: これらは NBD の代わりに RMI を用いたリモートディスクである。

RemoteDisk はクライアント、DiskServer はサーバである。

我々は VLSD を用いて 70TB ストレージを構築した。図 1 に概要を示す。

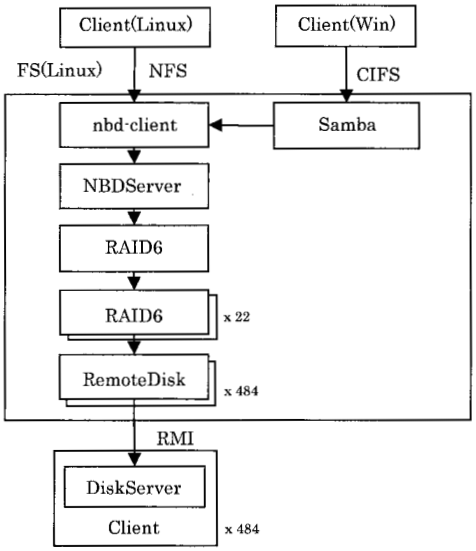


図 1　70TB システムの概要

このようなシステムの構成は以下のように行う。クライアントの利用状況は一様ではないので、未使用領域は複数のディスクに分散しているかもしれない。そこで、各クライアントにおいて未使用領域を JBOD で集めて 170GB 確保する。これを DiskServer で共有する。ファイルサーバでは、500 の RemoteDisk を作成し、22 ずつで RAID6 を構成する。さらに、22 の RAID6 で RAID66 を構成する。最後に RAID66 を NBDServer で共有する。NBD は安全なプロトコルではないためサーバ内部で局所的に用いる。

ファイルサーバには Linux マシンを使う。Windows クライアントには CIFS で、Linux クライアントには NFS で接続を許す。ファイルサーバには 1 つの NBD デバイスを作成し、NBDServer と接続する。このデバイス上に XFS を作成すればストレージは完成する。クライアントはそれぞれのプロトコルでアクセスすることができる。

## 4. セキュリティ

VLSD を用いて実際の大規模ストレージを運用するには、セキュリティの問題がある。具体的には、以下の問題がある。

- PC における盗聴・改ざん
- 通信路における盗聴・改ざん
- なりすまし

本節では、これらの問題を解決するセキュリティ技法について述べる。

### 4.1 概要

VLSD セキュリティフレームワークのクラスは以下の通りである。

AuthServer: 認証を行い、認証結果としてチケットを発行する。

SecureRemoteDisk: 安全な RemoteDisk。要求ごとにチケットを付加する。

SecureDiskServer: 安全な DiskServer。チケットを検証し、そのチケットが不正ならば要求を拒絶する。

CipherDisk: データを暗号化することでデータを保護する。書き込み時にデータを暗号化し、読み取り時にデータを復号化する。保存されたデータは暗号化されているため、仮想ディスクファイルを直接読み取られてもデータは保護される。

LogDisk: ディスクアクセスを記録する。ただし、キャッシュなどが遅延書き込みを行う場合、そのイベントは即時に記録されるとは限らない

ReadOnlyDisk: 書き込みを禁止することで不正なアクセスからデータを保護する。

UnionDisk: ReadOnlyDisk に対する書き込みを、異なる Disk に記録する。VariableDisk は、UnionDisk に NullDisk（サイズ無限、読み取りのみ、データはすべて 0）を組み合わせたものに等しい。保護したいディスクを ReadOnlyDisk で保護し、それに対する書き込みを UnionDisk に保管することで、効率よく ILM(Information Lifecycle Management)を実現する。

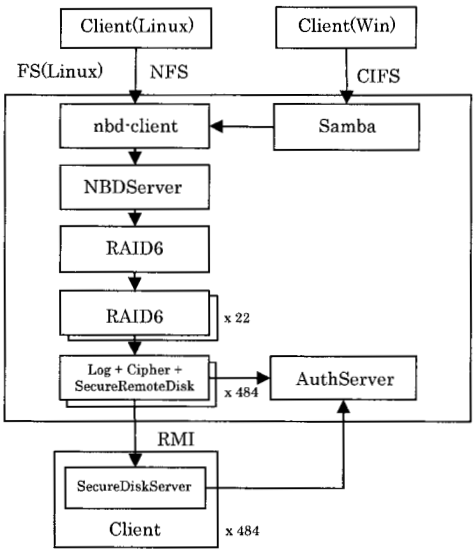


図 2　安全な 70TB システムの概要

VLSD のセキュリティは AAA アーキテクチャに基づく。AAA とは認証 (authentication)、認可 (authorization)、監査 (audit) を意味する。認証は AuthServer によって実現される。認可は SecureRemoteDisk, SecureDiskServer などによって実現される。原則としてファイルサーバが読み書きを代行するため、認可は認証と等しい。つまり、認証された利用者にはフルアクセスを許可する。監査は LogDisk によって実現される。

これらのクラスを用いて構成した 70TB システムを図 2 に示す。

### 4.2 PC における盗聴・改ざん

我々の大規模ストレージは、多くのクライアント PC の容量の一部を集めることで構築される。ある秘密の情報のアクセスを制限するとき、サーバ OS でアクセス権を正しく設定しても、クライアント上でディスクの内容を直接読み取ることで不正アクセスが可能となる。この問題には 2 つの原因がある。第 1 の原因はクライアント上のユーザが VLSD リモートディスクにアクセスできることである。第 2 の原因はクライアント上のユーザが VLSD リモートディスクの内容を読み取ることができることである。

第 1 の原因は OS のアクセス権を正しく設定することで解決できる。第 1 の原因が解決されたら、第 2 の原因も解決される。しかし、第 1 の原因を解決することが困難なことがある。例えば、あるユーザが自分の PC の HDD の一部をシステムに提供するとき、そのユーザは自分の PC の管理者であろう。管理者はすべてのファイルにアクセスできる。ゆえに、データを不正に改ざんできる。このような場合、第 1 の原因を解決することはできない。しかし、このような場合でも、第 2 の原因を解決すれば、データを保護することができる。例えば、他の PC と組み合わせて RAID を構成すれば、その PC のユーザだけが不正な改変を行ってもマスクできる。VotedRAID1 は多数決により任意故障をマスクできる。しかし、VotedRAID1 の性能はよくない。また別の例では、各書き込みの CRC を記録するように LogDisk を拡張する。そして、読み取り時に CRC をチェックすることで任意故障を検出するクラスを作成する。VLSD では、このようなカスタマイズが容易である。

第 2 の原因は、ディスクを暗号化することでも解決できる。第 2 の原因だけ解決する方法として、VLSD は CipherDisk を提供する。CipherDisk は、ディスクの内容を暗号化することで、それを保護する。

次に、改ざんを防ぐことができないとき、データを保護する方法について説明する。VLSD では、保護したいディスクを ReadOnlyDisk でラップすることで、書き込みを禁止することができる。書き込みする必要がある場合、UnionDisk を用いて保護したいディスクに対する差分を作成することができる。オフラインとなったときに、差分を検証して、必要なら適用すればよい。

### 4.3 通信における盗聴・改ざん

我々の大規模ストレージは、多くのクライアント PC の容量の一部を集めることで構築される。このため、クライアントがサーバと通信するフロントエンドと、サーバがストレージと通信するバックエンドが等しい。よって、クライアントが直接ストレージを不正にアクセスすることができる。この問題を解決するには、認可されたアクセスのみを許可する必要がある。

VLSD では、SecureRemoteDisk と SecureDiskServer の組で通信を保護する。SecureRemoteDisk を使うには、正当なチケットを必要とする。SecureRemoteDisk は、すべての要求メッセージにチケットを付加して送信する。SecureDiskServer は受信したチケットを検証する。不正なアクセスは拒否する。正当なアクセスのみ実行する。

### 4.4 なりすまし

認可に必要なチケットは AuthServer によって発行される。SecureDiskServer はチケットを検証するとき、AuthServer に依頼する。一度検証したチケットはキャッシュし、通信量を最適化する。

チケットは正当なアクセスを示すケーパビリティである。よって、チケットが漏洩すると、不正アクセスが可能となる。侵入者は通信パケットを盗聴することで、チケットを入手できる。よって、通信路を SSL または CipherDisk で保護する必要がある。VLSD にはデータを暗号化する CipherDisk がある。よって、VLSD だけでも通信路を保護することはできる。しかし、後述のように CipherDisk の性能はよくない。IPsec や SSL のほうが CipherDisk より効率がよい。

## 5. 評価

評価環境は以下の通りである。CPU は Pentium M 1.6GHz、メモリは 1GB、HDD は 60GB(ATA100)、OS は Windows XP SP2、Java は 1.6.0_01 である。

最初に、認証性能について述べる。AuthServer で認証するのに要する平均応答時間は 1.196771ms/req であった。これは十分短い。ただし、これは AuthServer 内部の DB を用いた場合である。AuthServer が外部 DB に問い合わせる場合、応答時間は DB に依存する。

次に、SecureDiskServer と SecureRemoteDisk の性能評価について述べる。SecureDiskServer、SecureRemoteDisk はそれぞれ DiskServer、RemoteDisk にチケットに基づく認可機能を加えたクラスである。これらのクラスを使うことで、正しいチケットを持つものだけがディスクにアクセスできる。SecureRemoteDisk/SecureDiskServer の性能を表 1 に示す。

表 1　SecureRemoteDisk の性能

| | RemoteDisk | SecureRemoteDisk |
|---|---|---|
| Read 1GB | 65.2s | 71.0s |
| Read/Write 1GB | 151.3s | 168.3s |

ここで、テストは 2 種類行った。1GB データの読み取り(Read)および読み書き(Read/Write)である。RemoteDisk は DiskServer と、SecureRemoteDisk は SecureDiskServer とそれぞれ通信する。RemoteDisk から 1GB のデータを読み取るとき、以下の処理が行われる。第 1 に、RemoteDisk が DiskServer に 8KB 単位で読み取り要求を送信する。第 2 に、DiskServer が FileDisk からデータを読み取る。第 3 に、DiskServer は読み取ったデータを RemoteDisk に返信する。SecureRemoteDisk も同様である。ここで、読み取りのオーバーヘッドは 6s、読み書きのオーバーヘッドは 17s であった。ゆえに、SecureRemoteDisk のオーバーヘッドは RemoteDisk に対して 10%であるといえる。

次に、CipherDisk の性能を評価する。CipherDisk はデータを暗号化することでデータを保護する。テストは 1GB データの読み書きである。CipherDisk の性能を表 2 に示す。

表 2　CipherDisk の性能

| クラス | 暗号法 | 応答時間 |
|---|---|---|
| FileDisk | | 50.3s |
| CipherDisk | AES/ECB/NoPadding | 148.3s |
| | DES/ECB/NoPadding | 336.4s |
| | DESede/ECB/NoPadding | 973.0s |

AES のオーバーヘッドは約 100s/GB である。暗号と復号の両方を行い、それらの処理時間が等しいと仮定すると、片方の処理時間は 50s/GB である。これは FileDisk の応答時間に等しい。つまり、AES に基づく CipherDisk は FileDisk より 3 倍遅い。DES は AES の 3 倍遅い。DESede は DES の 3 倍遅い。DESede は 3DES であるから、この結果は自然である。いずれにせよ、暗号化のオーバーヘッドは無視できない。ゆえに、CipherDisk を用いるより、ホスト OS のファイル保護機構を用いてデータを保護したほうがよい。

## 6. まとめ

本論文では、VLSD のセキュリティ機能について述べた。従来の VLSD ストレージシステムでは、なりすまし、改ざん、盗聴が可能であった。新たな VLSD を適切に用いることで、VLSD ストレージシステムでは、なりすまし、改ざん、盗聴を防ぐことができる。

本研究によってすべてのセキュリティ問題が解決されたわけではない。例えば、本論文で提案したセキュリティフレームワークでは、正しい VLSD がすべてのクライアントにインストールされていることを前提としている。VLSD 自体が改ざんされていることは考慮していない。


## 謝辞

本研究は科研費基盤(C)「PC グリッドによる高信頼・高効率な分散仮想ストレージの研究(19500066)」により援助されています。


## 参考文献


[1] Chai Eriant, Minoru Uehara, Hideki Mori: "A Case Study on Large-Scale Disk System concatenating Free Space", ICICIC2007 (2007.9.5-7)

[2] Chai Eriant, Minoru Uehara, Hideki Mori, Nobuyoshi Sato: "Virtual Large-Scale Disk System for PC-Room", LNCS 4658, Network-Based Information Systems, pp.476-485, (2007.9.3-4)

[3] David A. Patterson, Garth Gibson, and Randy H. Katz: "A Case for Redundant Arrays of Inexpensive Disks(RAID)", ACM SIGMOD, 1988

[4] Peter M. Chen, Edward K. Lee, Garth A. Gibson, Randy H. Katz, and David A. Patterson: "RAID: High-Performance, Reliable Secondary Storage," ACM Computing Surveys, Vol. 26, No. 2, pp.145-185, June 1994

[5] J. Wilkes, R. Golding, C.Stealin, and T. Sullivan: "The HP AutoRAID hierarchical storage system," ACM Trans. Computer Systems, 14(1), pp.108-136, Feb., 1996.

[6] Sung Hoon Baek, Bong Wan Kim, Eui Joung Joung, Chong Won Park: "Reliability and performance of hierarchical RAID with multiple controllers," In Proc. of 12th annual ACM symposium on Principles of Distributed Computing, pp.246-254, 2001

[7] Edward K. Lee, Chandramohan A. Thekkath: "Petal: Distributed Virtual Disks", http://www.thekkath.org/papers/petal.pdf

[8] Qin Xin, Ethan L. Miller, Thomas Schwarz, Darrel D. E. Long, Scott A. Brandt, Witold Litwin: "Reliability Mechanisms for Very Large Storage Systems," In Proceedings of the 20th IEEE/11th NASA Goddard Conference on Mass Storage Systems and Technologies (MSS'03), (2003)

[9] Osamu Tatebe, Noriyuki Soda, Youhei Morita, Satoshi Matsuoka, Satoshi Sekiguchi, "Gfarm v2: A Grid file system that supports high-performance distributed and parallel data computing," Proceedings of the 2004 Computing in High Energy and Nuclear Physics (CHEP04), Interlaken, Switzerland, September 2004

[10] Lustre a network clustering FS, http://www.lustre.org/

[11] Syuuichi Ihara: "Tokyo Tech Tsubame Grid Storage Implementation", (2007.5), http://wiki.lustre.org/images/7/79/Thumper-BP-6.pdf