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

2007－ARC－172 (19)
2007－HPC－109 (19)
2007／3／1

仮想クラスタに対するIPストレージの提供方法の比較

谷村勇輔† 小川宏高† 中田秀基†
田中良夫† 関口智嗣†

ユーティリティ・コンピューティングの1つとして，事前予約に基づいて仮想クラスタを貸し出すシステムの提案が行われている．貸し出す仮想クラスタでは計算機資源だけでなく，ストレージやネットワークを含めた仮想環境を提供することを目指しており，ストレージの仮想化とそれを仮想クラスタの各ノードへ提供する方法を検討する必要がある．本論文では，Xenによって構築した仮想計算機への仮想ストレージとして，仮想計算機をサービスする実計算機のローカルディスクを提供する方法に加え，NFSやiSCSIを用いて遠隔で管理されたディスクスペースを提供する方法を検討した．そして，データインテンシブなアクセスとメタデータインテンシブなアクセスに対するそれらの性能の違いを明らかにし，仮想クラスタでの利用に向けた課題を明らかにした．

Comparison of Methods for Providing An IP Storage to A Virtual Cluster System

Yusuke Tanimura,† Hirotaka Ogawa,† Hidemoto Nakada,†
Yoshio Tanaka† and Satoshi Sekiguchi†

A "virtual cluster" hosting system is proposed as an application of utility computing. In the system, all of three resources, computing, storage, and network resources, are virtualized to host customer's applications. This paper discusses how to virtualize the storage resource and provide it to the virtual cluster, which is composed by Xen-based virtual computers. Alternative methods to use of a local hard disk on each real computer, are use of a remote disk space on the central storage servers by NFS or iSCSI technology. The performance for data-intensive access and metadata-intensive access are compared among those methods, and the issues in using them for the virtual cluster are discussed.

1. はじめに

ユーティリティ・コンピューティングとはユーザが必要とする時に必要な量の計算機資源や計算機サービスを提供する方式であり，利用量に応じて課金が行われるビジネスモデルである．ユーティリティ・コンピューティングをデータセンタに応用すると，計算機を効率良く運用して利益の増加に結びつけるために，事前予約に基づいて資源の一部を貸し出すことが考えられる．我々は貸し出すシステムの構築を低コストで柔軟に行えるように仮想化を用いてそれを実現するアプローチをとっている[1]．運用の自由度や安全性を考えて，契約毎に貸し出す計算機資源の集合を仮想クラスタとしてまとめ，仮想クラスタの各ノードに対するストレージやネットワークも含めて仮想化することを目指している．

† 産業技術総合研究所
National Institute of Advanced Industrial Science and Technology

通常，仮想計算機に対するストレージの提供方法としてはホスト計算機のファイルシステム上のファイル，あるいはホスト計算機に接続されたディスクのパーティションを仮想計算機のディスクとして認識させることで実現する．しかし，この方法では仮想計算機のディスクがホスト計算機に束縛されてしまい，容量設定の自由度が低くなる．また，仮想クラスタとして提供することを考えた場合，ホスト計算機の障害時に仮想計算機をマイグレーションすることが難しくなる．さらに，ストレージ資源がすべてのホスト計算機のストレージに分散してしまうことは，ストレージ資源の管理という側面からも好ましくなく，専用のストレージノードに集約できることが望ましい．そこで，我々はネットワーク化されたストレージを仮想クラスタに提供する方法を検討する．ただし，専用機器を用いてSANを構成することは仮想化による利点を損なうため，ネットワーク・ストレージの中でもIPストレージを検討の対象とする．特に，iSCSI[2]は遠隔の計算機に対してブロックレベルのアクセスを提供するため，個々の仮想計算機へのディスク提供を管理しやすく，

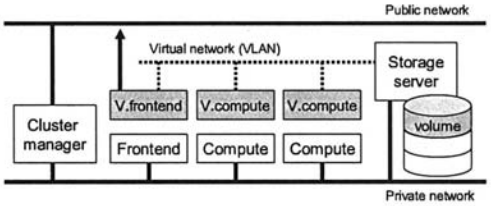

図 1　構築する仮想クラスタのアーキテクチャ

興味深い技術の 1 つである．

本研究では，Xen[3] を用いて構築した仮想計算機に対して，ストレージサーバ上のディスクスペースを NFS，あるいは iSCSI を利用して提供する方法と，ホスト計算機のローカルディスクを提供する方法との性能の違いを明らかにする．Xen により仮想化されたそれぞれのディスクスペース（仮想ストレージ）に対して，ファイルのメタデータの更新速度を測定するために PostMark v1.5，単一ファイルへの Read，Write のスループットを測定するために Bonnie++ v1.03 を用いる．こうして，仮想クラスタのストレージをネットワーク上に用意することの妥当性や，その実現方法において iSCSI の有効性を検証する．それらの結果を踏まえて，それぞれの仮想計算機に対する仮想ストレージをさらに仮想クラスタ内で共有するための仕組みについて検討する．

2. 仮想クラスタのアーキテクチャ

構築する仮想クラスタはおおよそ**図 1** に示すアーキテクチャとなる．図中のクラスタマネージャは，物理クラスタおよび仮想クラスタ群を管理するノードであり，資源予約に基づいて，予約時間が始まる前に仮想クラスタのセットアップを開始する．クラスタマネージャによって，仮想化されたフロントエンドノードと計算ノードがセットアップされ，それらは VLAN を用いてプライベートなネットワークを構成する．そして，ストレージ専用のサーバがすべての仮想計算ノード，あるいはいくつかの仮想計算ノードに対して，IP ストレージを用いてディスクスペースを提供することになる．ストレージサーバは大容量かつ信頼できるディスクスペースを備えており，それを論理ボリュームとして動的に切り出せるシステムであることを想定している．

中田らによって提案されている構築ツールは，クラスタのデプロイシステムである NPACI Rocks を用いて，このアーキテクチャに基づく仮想クラスタを容易に，かつ短時間で構築するためのものである．そのツールでは VMware あるいは Xen による仮想計算機をセットアップする[1]．

3. 仮想ストレージの提供方法

本研究では Xen を利用した仮想計算機へのディスク・ストレージの提供方法について検討する．Xen はその上で複数のゲスト OS を実行できる仮想マシンモニタであり，実際のハードウェアを完全にエミュレートするのではなく，部分的な仮想化やハードウェアによる仮想化を利用して性能の良い仮想環境を提供する[3]．Xen ではゲスト OS の 1 つが全てのハードウェアに制限なくアクセスできるドメイン 0 となり，それ以外のドメイン U で動くゲスト OS に対するハードウェア資源の割り当てや破棄を指示する．ディスクも同様に，Xen により仮想ブロックデバイス（VBD）として実装されている．ドメイン 0 では各物理ディスクに対応した VBD が提供され，それらに自由にアクセスできる．しかし，ドメイン U では，ドメイン 0 においてドメイン U 用に作成された VBD にのみアクセス可能である．ディスクアクセスが発生すると，ドメイン U の VBD ドライバは Xen に直接アクセスする．VBD に対応する物理ディスクスペースのアクセス権限が Xen によって確認された後，ドメイン U でピンされたメモリと物理ディスク間でのゼロコピーの DMA 転送が行われる．

ドメイン U への VBD の提供方法としては，ファイルイメージを利用する方法（File-backed VBD，以降 f-VBD と記す）と物理パーティションを直接利用する方法（Physical device VBD，以降 p-VBD と記す）の 2 種類がある．Xen v3.0.3 では，ドメイン 0 に Blktap ドライバを導入することで両者を tapdisk として抽象化し，全てのブロック要求をゼロコピーの非同期 I/O として処理するようになっている．

f-VBD と p-VBD を仮想計算機サーバ上のハードディスク，または NFS や iSCSI を利用してストレージサーバ上のハードディスクで提供する手法を考えると，**図 2** に示すようにドメイン U に対して 7 種類の仮想ストレージの提供方法を考えることができる．

file-local　ドメイン 0 において，ローカルのハードディスクに保存されたファイルイメージを利用して f-VBD を提供する方法．

phy-local　ドメイン 0 において，ローカルのハードディスクのパーティションをそのまま利用して p-VBD を提供する方法．

file-nfs　ドメイン 0 がストレージサーバ上のファイルイメージを NFS でアクセスして f-VBD を提供する方法．

file-iscsi　ドメイン 0 がストレージサーバ上の論理ボリュームを iSCSI でアタッチして，そこに保存されたファイルイメージを利用して f-VBD を提供する方法．

phy-iscsi　ドメイン 0 がストレージサーバ上の論理

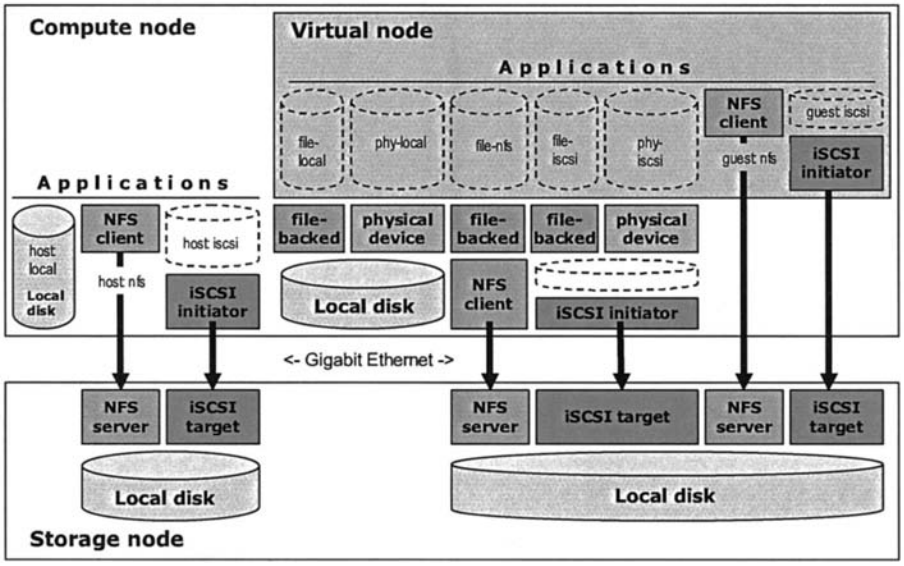

図 2 仮想ストレージの提供方法

表 1 測定環境

Machine spec.	CPU: Xeon 2.8 GHz x 2, Memory: 1GB 1.6 TB storage (RAID0 by 8 disks) - RAID controller: 3ware 7500-8 - Disk: Maxtor 6Y200P0 (ATA/133) NIC: Intel PRO/1000 (82546EB) OS: Fedora Core 6
Xen	v3.0.3 - Domain0 kernel: xen-2.6.18-1.2869.fc6 - DomainU kernel: 2.6.18-1.2798.fc6xen
iSCSI	Initiator: Open-iSCSI v2.0-747 Target: iSCSI Enterprise Target v0.4.14
Network switch	Force10 E600

ボリュームを iSCSI でアタッチして，そのまま p-VBD として提供する方法.

nfs ドメイン U がストレージサーバ上のディレクトリを NFS で直接マウントする方法.

iscsi ドメイン U がストレージサーバ上の論理ボリュームを iSCSI で直接アタッチする方法.

4. 性能比較

前節で説明した 7 種類の仮想ストレージの提供方法について，データインテンシブなアクセスパターンとメタデータインテンシブなアクセスパターンに分けて性能測定を行った．**表 1** に記したスペックのマシンを 2 台用意し，片方を仮想計算機サーバ，他方をストレージサーバとしてセットアップし，1Gbps のネットワークで接続した．Xen 上ではドメイン 0 の他に，Para-virtulized されたドメイン U を 1 つだけ動かす．

表 2 NFS および iSCSI のパラメータ

NFS	Mount options: rw, vers=3, rsize=32KB, wsize=32KB, hard, udp, timeo=11, retrans=2. Ext3 filesystem on the server.
iSCSI	InitialR2T: Yes MaxRecvDataSegmentLength: 64KB MaxXmitDataSegmentLength: 8KB FirstBurstLength: 64KB MaxBurstLength: 256KB MaxOutstandingR2T: 1

CPU はドメイン 0 とドメイン U にそれぞれ 1 つずつピンされるように設定し，ドメイン 0 のメモリ割り当ての上限，およびドメイン U のメモリ割り当ては 512MB とした．ドメイン U のネットワークは，ドメイン 0 およびストレージサーバが接続するネットワークにブリッジ接続させた．NFS，iSCSI で利用するパラメータは**表 2** の通りである．

ストレージに関しては，両方のサーバにおいて RAID0 で構成された 1.6TB のディスクの一部を LVM (Logical Volume Manager) の管理下に置いた．LVM で管理されたボリュームから，仮想計算機サーバではローカルに f-VBD と p-VBD として割り当てる領域を切り出し，ストレージサーバでは NFS あるいは iSCSI 経由で提供する f-VBD や p-VBD，あるいはドメイン U から直接アクセスされるディスク領域を切り出す．p-VBD 用に切り出す領域は 10GB，それ以外の用途で切り出す領域は 40GB とした．そして，f-VBD を提供する場合は，切り出した領域において 10GB のファイルイメージを作成した．ファイルイメージの作成に関して，Xen のユーザマニュアル[4] では Sparse

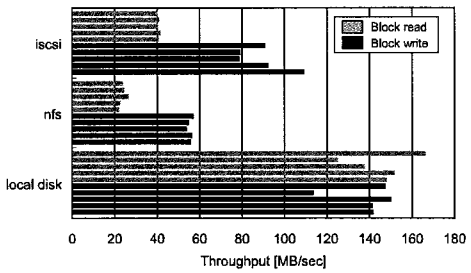

図 3 ドメイン 0 での Sequential I/O の性能

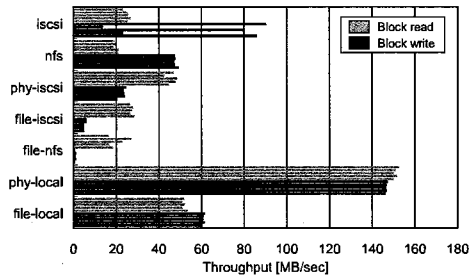

図 4 ドメイン U での Sequential I/O の性能

ファイルを作成するよう紹介されているが，4.1 節と 4.2 節の実験では Sparse ファイルは利用せず，ファイル作成時にファイルイメージ全体をゼロ埋めしている．Sparse ファイルを利用した場合の性能は 4.3 節に記す．また，本実験では全ての方法において Ext3 ファイルシステムをデフォルトのパラメータで利用した．

4.1 データインテンシブなアクセス

データインテンシブなアクセスの性能測定には Bonnie++ v1.03 を利用した．ブロックサイズを 8KB に設定し，1GB のファイルの Sequential Block Read, Sequential Block Write の性能を測定した．**図 3** はドメイン 0 のローカルディスク，およびドメイン 0 からストレージサーバ上のディスクに対して同様の実験を行った結果であり，ドメイン U の結果を分析する上での予備評価である．そして，ドメイン U の実験結果は**図 4** となった．測定は 5 回行い，それぞれの値を棒グラフで示している．ドメイン U の実験においては，表 2 のパラメータではドメイン U から直接 NFS マウントを利用する場合に，Bonnie++のプログラムが 1 日待っても終了しない状況に陥った．そのため，この方法においてのみ，NFS のマウントオプションで指定するプロトコルを UDP から TCP に変更して実験を行った．

結果より，phy-local はローカルディスクの性能を得ることができているが，file-local ではローカルディスクの半分以下の性能しか得られていない．file-nfs や file-iscsi もドメイン 0 での NFS や iSCSI のアクセスに比べて低い性能しか示さず，Write 性能は著しく低い．phy-iscsi については，Read は file-local より若干低い程度の性能を示しているものの phy-local と比べると 1/3 の値であり，Write 性能は 1/5 の値である．ドメイン U において直接 NFS や iSCSI を利用した方が VBD 経由でアクセスするよりも良い性能が得られている．ただし，ドメイン U の iSCSI の性能は安定しているとは言えない．

表 3 PostMark のパラメータ

Number of transactions	100,000
Block size	8 KB
Number of files	1,000
Number of subdirectories	100
Read / append ratio	50 % / 50 %
Create / delete ratio	50 % / 50 %

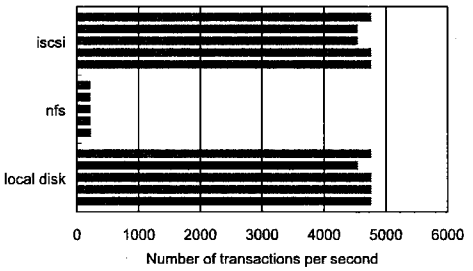

図 5 ドメイン 0 でのファイル操作の性能

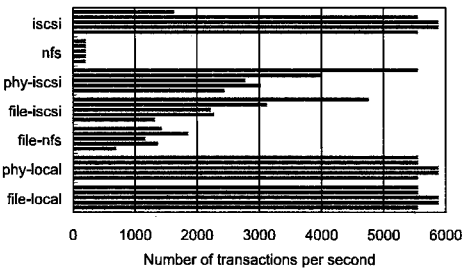

図 6 ドメイン U でのファイル操作の性能

4.2 メタデータインテンシブなアクセス

メタデータインテンシブなアクセスの性能は PostMark v1.5 を用いて計測した．PostMark は比較的小さいサイズのファイルを指定された数だけ作成し，Read, Append, Create, Delete の操作をランダムに行うシングルスレッドのベンチマークである．本実験では PostMark のパラメータとして**表 3** を設定し，メタデータの更新頻度が高いアクセスを行った．

ドメイン 0 での予備実験の結果を**図 5** に，ドメイン U での実験結果を**図 6** に示す．ドメイン U では file-local と phy-local の性能差がほとんどなく，それ

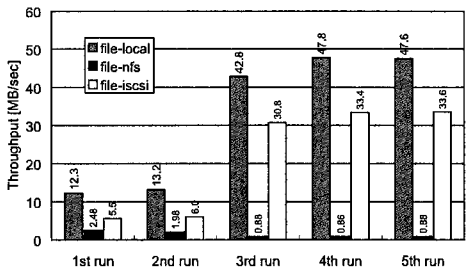

図 7 Sparse ファイルベースの仮想ディスクに対する書き込み性能

らはドメイン 0 での local disk の結果より良い値を示した．一方，ドメイン 0，U での iscsi はローカルディスクを利用する方法と同等の性能を示した．最良値で比較すると，phy-iscsi や file-iscsi も phy-local と遜色のない値であるが，試行毎の結果から分かるように安定した性能が得られていない．nfs の性能はいずれのドメインでも低い結果となったが，file-nfs はそれらより良い性能を示した．これは 1 つのファイルが NFS 経由でアクセスされることによる影響と考えられる．

4.3 Sparse ファイルが性能に与える影響

Sparse ファイルを利用するとディスク利用の初期の段階においてオーバーヘッドが考えられる．その影響を調査するべく，ドメイン U から Sparse ファイルベースの f-VBD を利用した時の書き込み性能を測定した．まず，2GB の Sparse ファイルを作成し，Ext3 でファイルシステムを作成した．その時点での実サイズは 98MB である．次にそれを f-VBD としてドメイン U から利用できるように設定し，8KB のブロックサイズで Block Write を行い，1GB のファイルを作成するプログラムを実行した．1GB のファイルを作成後，そのファイルを削除し，再度同じプログラムを動かした．

5 回目までのプログラムの実行における Write スループットを**図 7** に示す．図 7 から分かるように，データがファイルシステムに書き込まれていない状況では，書き込み性能は file-local で 25%，file-iscsi で 18%にまで低下した．しかし，file-nfs の場合は Sparse ファイルを利用する方が 2 倍以上性能が良い．これはファイルイメージのサイズが小さい時に NFS へのアクセスが速くなっているためと考えられる．

4.4 考　　察

IP ストレージを利用して，Xen による仮想計算機に対して f-VBD や p-VBD を提供する方法は，ホスト計算機のローカルディスクを利用する方法に比べて，十分な性能が出ているとは言えない結果が得られた．その原因として，Xen では VBD のバックエンドがネットワーク・ストレージである場合に性能が引き出せない設計や実装になっている可能性が考えられる．図 4 において iscsi，phy-local の Write 性能に比べて phy-iscsi の Write 性能が著しく悪いことや，iscsi と file-iscsi や phy-iscsi とで Read/Write 性能が逆転していることがその可能性を示している．一方，Derek らによる Position Paper[5] では，iSCSI 用の Virtual Channel Processor（VCP）を用意して，iSCSI のトラフィックをストリームとして実計算機の NIC に流すような設計が検討されており，性能改善の可能性についても言及されている．

file-nfs は file-iscsi や phy-iscsi に比べて，いずれの実験においても低い性能であった．NFS は複数の計算機でファイル共有を可能にすることを目的としており，一貫性を保つためにメタデータの同期を頻繁に行う必要があり，RPC に基づくプロトコルを採用している．そのため，今回の用途では iSCSI との設計の違いによる性能差が現れた．

ドメイン U から直接 iSCSI を利用する方法は比較的良い性能を示すこともあるが，安定していない．今回の実験で得られたような性能差であれば，仮想計算機側に iSCSI のソフトウェアをインストールし，設定を行う必要があることのデメリットの方が大きい．なぜなら，我々はユーザに root 権限を与えて仮想クラスタを貸し出すことを考えており，こうした設定を仮想クラスタのユーザから隠蔽したいためである．

性能面と管理面の双方を考慮すると，ストレージサーバ上のディスクスペースを仮想計算機に提供する方法として，今回比較した方法の中では phy-iscsi が妥当な選択である．ただし，phy-local と同等の性能が安定して得られない原因の調査とそれに基づく性能改善は重要な課題である．また，f-VBD のファイルイメージを別の計算機に移動するなどのポータビリティを考慮すると，file-iscsi を第 2 の選択肢として考えることができる．この場合には，Sparse ファイルを利用する際に書き込み性能が著しく低下する可能性があることに注意が必要である．

5. 仮想ノード間のデータ共有スペースの提供に向けて

仮想クラスタへのストレージの提供については，個々の仮想ノードに対するストレージの提供だけでなく，仮想クラスタのノード間でプログラムやデータを共有するための仕組みも検討する必要があり，今後の課題である．データ共有スペースを実現する方法は通常のクラスタでも多様なツールが利用されているため，本研究ではユーザがアプリケーションの特性を考えて任意の共有ファイルシステムを選択し，それを仮想クラスタの構築ツールが事前にセットアップする方法を設計している．

ここでは 4 節の結果を踏まえて，iSCSI ベースの仮想ストレージを利用して共有スペースを実現する方法を下記に 3 つ挙げる．IP-SAN や Parallel Filesystem

ではストライピング機能を利用することで全体のI/Oスループットを上げることができるため，iSCSIを用いた仮想化の性能面の課題を補うという使い方も考えられる．

NAS Head NASヘッドとなる仮想ストレージサーバを1つ用意し，iSCSIを用いて実ストレージサーバ上の論理ボリュームをアタッチする．そして，それを仮想計算ノードがNFSを用いてマウントすることでデータ共有を行う．この方法の欠点はNASヘッドが性能のボトルネックになる可能性が高いことである．

IP-SAN Red Hat GFS2[6]やLustre[7]などSANに基づくクラスタファイルシステムを用いて，仮想計算ノードがそれぞれiSCSIでアタッチしたストレージサーバ上の論理ボリュームを仮想クラスタ全体で共有する．ただし，仮想メタサーバあるいは仮想ロック管理サーバを別途用意する必要がある．この方法の利点は，要求される性能や予算に応じてIP-SANを構築するだけでなく，高価なSANインフラストラクチャも同様のアーキテクチャで利用可能であることである．もちろん，後者を利用する場合はストレージやネットワーク資源も含めた完全な仮想クラスタではなくなる．

Parallel Filesystem TCP/IPベースの並列ファイルシステムであるPVFS2[8]やGfarm[9]を利用してデータ共有を行う．前者はMPI-IOが利用可能であり，後者はデータアクセスの局所性を活かした並列処理が可能である．ユーザがそうしたアプリケーションの特性を考慮して利用する必要がある反面，低コストだが性能の良いデータ共有を実現する．IP-SANの方法と同様，この方法においても仮想メタサーバが別途必要になる．

これらの実現に向けては，仮想クラスタ構築ツールにおいて，特殊なカーネルをゲストOSで利用できるようにすることや，仮想計算ノード以外の機能をもつ仮想サーバ（仮想メタサーバなど）をセットアップする機能のモジュール化が必要である．さらに，仮想計算ノードの数が増加すると，iSCSI Targetの役割を果たすストレージサーバへの負荷の集中が起きることが予想される．これはストレージサーバを複数用意することでおおよそ対応可能であると考えられるが，1台のTargetノードで何台のiSCSI Initiatorにレスポンス良く対応できるかを検討しなければならない．ユーザ向けには，データ共有あるいはデータのアクセスパターンに応じて，それぞれの方法における性能の指標を提示することが重要である．

6. まとめ

本研究では仮想クラスタに対してストレージ資源をどのように仮想化し，提供することが可能であるかを検討した．Xenによる仮想計算機単体に対して7種類の方法を検討し，ストレージサーバを利用してストレージ管理を集中的に行う場合には，iSCSIでアタッチした論理ボリュームを用いてp-VBDを構成する方法（phy-iscsi）が有効であることを確認した．しかし，ホスト計算機のローカルディスクを利用するp-VBDやiSCSIを単体で利用する時の性能と比べて，phy-iscsiの性能が十分に出ていないという課題が明らかとなった．そして，4節で示した実験結果を踏まえて，5節ではiSCSIで提供した仮想ストレージをさらに仮想ノード間で共有する仕組みについて述べた．

今後は今回明らかになったiSCSIベースのVBDに関する性能面の課題を解決していくとともに，ファイル共有の仕組みやiSCSIを利用するノード数の増加に対するストレージサーバのスケーラビリティについて検討を進めていく．

参考文献

1) 中田秀基, 横井威, 関口智嗣. Rocksを用いた仮想クラスタ構築システム. 情報処理学会研究報告 2006–HPC–106, pp. pp.49–54, 2006.
2) RFC 3720 - Internet Small Communication Systems Interface (iSCSI). http://www.ietf.org/rfc/rfc3720.txt.
3) P. Barham, B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, R. Neugebauer, I. Pratt, and A. Warfield. Xen and the Art of Virtualization. In *Proceedings of the 9th ACM Symposium on Operating Systems Principles (SOSP)*, 2003.
4) Xen v3.0 Users' manual. http://www.cl.cam.ac.uk/research/srg/netos/xen/readmes/user/.
5) D. McAuley and R. Neugebauer. A case for Virtual Channel Processors. In *Proceedings of the ACM SIGCOMM 2003 Workshop*, 2003.
6) Red Hat Global File System. http://www.redhat.com/software/rha/gfs/.
7) Lustre. http://www.lustre.org/.
8) PVFS2. http://www.pvfs.org/.
9) O. Tatebe, Y. Morita, S. Matsuoka, N. Soda, and S. Sekiguchi. Grid Datafarm Architecture for Petascale Data Intensive Computing. In *Proceedings of the 2nd IEEE/ACM Symposium on Cluster Computing and the Grid (CCGrid)*, pp. pp.102–110, 2002.