2007－HPC－111 (40)
2007／8／3

複数サイトにまたがる仮想クラスタの構築

広渕崇宏† 谷村勇輔† 中田秀基†
田中良夫† 関口智嗣†

計算機資源の効率的かつ柔軟な運用を実現する手法として，仮想クラスタ技術が注目されている．しかし既存システムは単一サイトの計算機資源のみを仮想クラスタの対象とする．構築可能な仮想クラスタは単一サイト内に存在する計算機資源によって制限され運用の柔軟性に乏しい．そこで，我々は複数サイトをまたいで横断的に仮想クラスタを構築可能な管理システムの実現に取り組んでいる．複数サイトにわたって構築された仮想クラスタにおいても，単一実行環境としての透過性やシステムの導入や運用における容易性が実現されなければならない．本論文では，インターネットを介したイーサネットVPNによって単一のネットワークセグメントを仮想クラスタに対して提供することを提案する．予備的な評価実験においては，イーサネットVPNで結ばれたサイト間において仮想クラスタの構築が可能であることを確認した．

A Multi-Site Virtual Cluster with VPN

Takahiro Hirofuchi ,† Yusuke Tanimura ,†
Hidetomo Nakada ,† Yoshio Tanaka † and Satoshi Sekiguchi†

We are now developing an advanced cluster management system for multi-site virtual clusters; which provides a virtual cluster composed of distributed computer resources over wide area networks. It has great advantages over other cluster management systems designed for only single-site resources; users can create a cluster of virtual machines from local and remote physical clusters in a scalable manner, and dynamically change the number of cluster nodes on demand and seamlessly. In our system, a multi-site cluster needs to achieve a monolithic system view of cluster nodes to enable existing applications to be deployed quickly and managed flexibly as in physical clusters. In this paper, we propose to exploit Ethernet VPNs to bridge distributed cluster nodes over the Internet for a multi-site virtual cluster. It transparently allows a single network segment for virtual cluster nodes thereby hiding underlying network topology. Our experiments showed that virtual clusters were successfully installed through an Ethernet VPN between two remote sites.

1. はじめに

計算機資源の集中的な管理や効率的な運用を実現する手法として，仮想クラスタ技術が注目されている．計算機資源を仮想計算機技術や論理ストレージ技術などによって抽象化することで，必要に応じて動的な資源提供を可能にする．また多様なアプリケーションに対する運用環境全体をオペレーティングシステムを含めて柔軟に構築できる．

しかし，このような仮想クラスタシステムが，単一の計算機センタやデータセンタなどにおいて閉じて運用されるのみであれば，計算機資源の効率的な運用やその柔軟性において限界が存在するといえる．単一サイト内の限られた計算機資源によって，提供可能な仮想クラスタ規模が制限されてしまう．また，停電や災害時など単一サイトの大部分に影響するハードウェアの停止等においては，仮想クラスタの運用が困難になる．

したがって，複数サイトにまたがって横断的に仮想クラスタを構築し，状況に応じてその構成を動的に変更可能であることが望まれる．例えば，複数サイトの計算資源を必要に応じて予約することで，提供サービスの規模にあわせて，単一サイト内の計算機資源規模に縛られずに動的に仮想クラスタ数を増減可能にする．また必要に応じて仮想クラスタ全体を停止することなく他サイトに移動できることを視野に入れる．

本稿では，我々が先に提案した単一サイト向けの仮想クラスタ管理システム[11]をもとにした，複数サイトにまたがる仮想クラスタシステムの構築手法について述べる．

既存の仮想クラスタ管理システムを複数サイト化するにあたっては，その仕組みがユーザにとって適度に

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

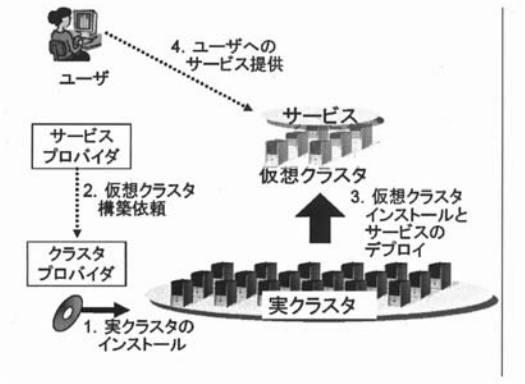

図 1 仮想クラスタ管理システムの利用シナリオ

透過的であり, 単一サイト内での仮想クラスタと同様のユーザビリティを提供しなければならない. さらにその拡張機構が既存システムと十分な親和性を有した手法である必要がある.

まず, 我々が開発中の単一サイト向けの仮想クラスタ管理システムの概要について述べ, 複数サイトへの拡張手法について検討する. 次に, システムの設計方針について述べ, その予備的な評価結果を報告する.

2. クラスタ管理システム

我々が開発中の仮想クラスタ管理システムは, 仮想マシンモニタによる計算機自体の仮想化, LVM および iSCSI[8] によるストレージの仮想化, および VLAN[3] によるネットワークの仮想化から構成される.

クラスタ管理システムは, 物理クラスタを構成する各計算機上に仮想マシンモニタを導入して仮想計算機を随時作成する. ストレージサーバから論理ストレージ領域を切り出して, iSCSI により仮想計算機にストレージを提供する. 各物理クラスタ上の仮想計算機を, ホスト OS のイーサネットブリッジを経由して, 仮想クラスタ固有の VLAN ID を付加した上で物理クラスタ間を結ぶローカルネットワークに接続する.

さらに, これら仮想化された計算機とストレージおよびネットワークの集合をひとつの仮想クラスタとして管理・運営するためのユーザインタフェースを提供する. クラスタを構成する仮想計算機の台数や, そのメモリおよびストレージ容量, 使用開始/終了時間などを指定して, 動的に仮想クラスタの作成が可能である. また, このとき仮想クラスタ内部に導入する OS やアプリケーションの指定も可能である. **図 1** に利用シナリオを示す.

システムの実装は, SDSC (San Diego Supercomputer Center) にて開発されたクラスタインストールツールの Rocks[7] をベースに, VMware Server による仮想化機構, Linux の iSCSI イニシエータおよびターゲット実装などを, 我々独自の管理システムの下で組み合わせている.

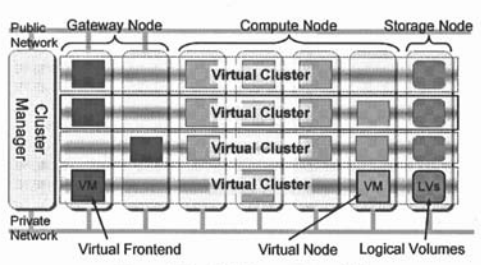

図 2 クラスタ管理システムの構成

システム構成は**図 2** に示される. 物理クラスタの各ノードは 4 種類存在する. **クラスタマネージャノード**は管理インタフェースを提供し, 要求に応じて仮想クラスタの作成や破棄などを他の物理ノードに対して命令する. また, Rocks におけるフロントエンドとして動作し, PXE ブートサーバ機能により, 他の物理ノードを自動的に設定する. **ゲートウェイノード**および**計算ノード**上では仮想マシンモニタが存在して複数の仮想計算機が動作し, それらはクラスタマネージャの要求に応じて作成される. 本システムでは仮想クラスタ内部のシステムインストールにおいても Rocks を用いる. そのため, ゲートウェイノード上で作成された仮想計算機は仮想クラスタ内部において Rocks におけるフロントエンドとして動作する. つまり生成された仮想クラスタにおける管理ノードとなり, 仮想クラスタ内部に対して PXE ブートサーバ機能や NAT 機能を提供する. **ストレージノード**ではクラスタマネージャの要求に応じて論理ストレージ領域を作成し, iSCSI ターゲットサーバから仮想クラスタに対して提供する.

ひとつの仮想クラスタを構成する各仮想計算機および論理ストレージ領域は, 他の仮想クラスタを構成するそれらとは完全に独立して取り扱われなければならない. そのため, 仮想クラスタごとに独立した VLAN を設定して, 各仮想計算機とそのストレージサービスが仮想的に独自のネットワークセグメント内に存在するよう工夫されている☆.

3. 仮想クラスタの複数サイト化

3.1 要求事項

先述したクラスタ管理システムをさらに拡張し, 複数サイト間で横断的に構築された仮想クラスタ (**図 3**) に対する要請を検討する.

第一に, 複数サイトにまたがって作成された仮想ク

☆ 本稿執筆時点ではホスト OS が iSCSI イニシエータとして働きストレージサービスは VLAN による管理下にない. しかし将来的に VLAN によってストレージサービスも分離してゲスト OS が直接 iSCSI イニシエータ機能をもつ選択肢も検討されている.

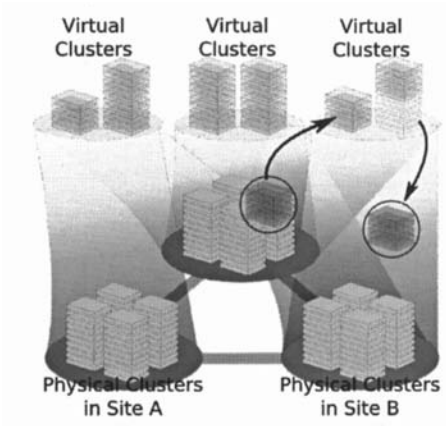

図 3 複数サイトからなる仮想クラスタ概念図

ラスタが引き続きユーザにとって透過性を有する必要がある．つまり，仮想クラスタを構成する仮想計算機等の物理的な配置にかかわらず，ひとつの仮想クラスタとして単一のシステムイメージとして見え続けなければならない．複数サイトにわたって作成された仮想クラスタも，一般的な物理クラスタと同様に任意のアプリケーションによる利用を妨げるものであってはならないと考える．

第二に，仮想クラスタ内部の OS やアプリケーションのインストールや設定が，容易かつ迅速に完了する必要がある．複数サイトによる大規模な仮想ノードからなるクラスタを構築する際にも，初期導入や管理のコストが増大することなく，運用を開始できなければならない．既存の成熟しつつあるクラスタ管理ツールを複数サイトにまたがる仮想クラスタ構築に際しても適用できなければならない．

第三に，仮想クラスタを構成する仮想計算機の動的な再配置を可能する必要がある．仮想クラスタを一切停止することなく仮想計算機の動的な再配置ができなければならない．たとえ仮想クラスタを構成する一部の仮想計算機を他のサイトに再配置したとしても，仮想クラスタ内部の OS やアプリケーションはその間引き続いてサービスを提供し続けられることが望ましい．

3.2 仮想クラスタネットワークの拡張

以上より，ひとつの仮想クラスタを構成する仮想計算機やストレージサービスを接続するネットワークは，複数サイトにまたがって仮想クラスタが構築された場合にも，依然としてひとつのネットワークセグメントとして存在すべきであると考える．

これにより，マルチキャストやブロードキャストを用いるアプリケーションを変更せずとも複数サイトにまたがった仮想クラスタに用いることができ，Rocks 等のクラスタインストールツールが引き続き利用可能になる．また，Xen[1] や VMware Infrastracture[9] などの仮想マシンモニタが備える仮想計算機のライブマイグレーション機能を使えば，仮想クラスタの動的な再配置も比較的容易に実現可能であると考える．

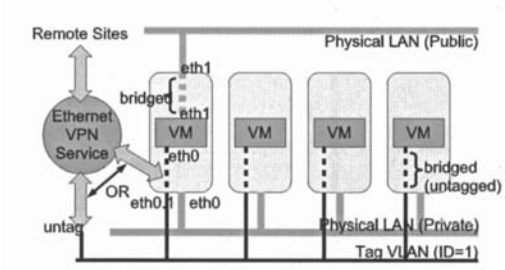

図 4 VPN による仮想クラスタの拡張

3.3 設計指針

複数サイトに対応した仮想クラスタ管理システムの概要を示す．既存の仮想クラスタ管理システムをサイト間で相互に仮想クラスタの構築を依頼できるよう拡張し，各サイトごとに作成された仮想クラスタをイーサネット VPN によってひとつの仮想クラスタとして統合して運用可能にする．

クラスタマネージャを拡張し，他のサイトのクラスタマネージャに対して仮想クラスタの構築を要請できるよう変更し，また逆に他のサイトからの要請も処理可能にする．このとき，同時にネットワーク帯域なども必要に応じて予約する場合もある．

次にサイトごとに構築された仮想クラスタを相互に接続するために，イーサネット VPN サービスを仮想クラスタごとに導入する (**図 4**)．サイト内において各仮想クラスタに対して割り当てられたネットワークは，タグ付き VLAN によって仮想クラスタごとに分離されている．VLAN タグは仮想計算機のホスト OS によって付与され，物理クラスタを構成する計算機間の LAN において VLAN タグ付きのイーサネットフレームが流れている．VLAN ID は 4096 個とその数に限りがあり，複数サイト間で使用可能な VLAN ID を一貫して管理することは避けたい．そこで，VPN 上で無駄なトラフィックを転送しないためにも，サイト間でまたがって構築された仮想クラスタにおいては，そのプライベートなネットワークにおけるイーサネットフレームのみを VLAN タグを取り外した上で対向の VPN サーバへ転送する．

イーサネット VPN の導入においては複数の手法が考えられる．商用の広域イーサネットサービスを用いたり，あるいは L2TPv3[4] や EtherIP[2] 等に対応したアプライアンスを利用できる．この場合，VPN 装置は物理クラスタ間の LAN に対して接続される．しかし，デメリットとして VLAN タグの取り扱いが煩雑であり，VPN 接続をクラスタマネージャから操作しにくい．

一方，OpenVPN[5] や Vtun[10]，PacketiX[6] などのソフトウェア VPN も利用できる．クラスタマネージャ

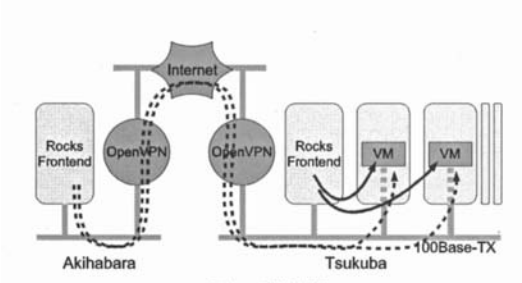

図 5 実験環境

表 1 実験に用いた計算機

つくば	
いずれも	Intel Xeon 2.80GHz 1GB RAM
秋葉原	
Rocks フロントエンド	Intel Core2 6300 2GB RAM
OpenVPN ノード	Intel Pentium4 3.40GHz 2GB RAM

との連携が比較的容易である．また，ゲートウェイノード上で仮想クラスタごとに複数の VPN サービスを動作させることで，VLAN ネットワークインタフェースから VLAN タグなしのイーサネットフレームを取り出し転送可能になると考えられる☆．予備評価においては導入が比較的容易な OpenVPN を用いている．

4. 予備評価

イーサネット VPN を用いて複数サイトにまたがる仮想クラスタを構築する上での予備的な評価実験を行った．

実験環境を**図 5** および**表 1** に示す．つくばおよび秋葉原に開発中の仮想クラスタ管理システムを導入し，両者をそれぞれインターネットに接続する．さらに OpenVPN をインストールした VPN 専用ノードもそれぞれに用意する．OpenVPN はブリッジ VPN として設定し，つくば・秋葉原それぞれの LAN で流れるイーサネットフレームがすべて互いに転送される状態になっている．

ping および iperf による計測により，本実験中につくば・秋葉原間では往復遅延 4.0ms 程度，また可用帯域 95Mbps 程度となっている．また VPN 経由で計測すると，デフォルトの Blowfish-CBC によるデータ暗号化を用いた場合で，往復遅延 4.2ms 程度，可用帯域 89Mbps 程度である．このとき VPN 専用ノードでの CPU 負荷は低いままとなっている．データ暗号化なしの場合も，CPU 負荷がさらに低くなる以外は同様の結果を得ており，十分な性能を有する計算機を OpenVPN ノードとして用いる限り，暗号化のオーバヘッドは無視できることを確認した．

☆ 例えば eth0.1 等の仮想クラスタごとの VLAN ネットワークインタフェースを直接イーサネット VPN の対象とできる．

4.1 仮想クラスタ構築時間

次に VPN を介して仮想クラスタを構築する際のオーバヘッドを計測した．つくば側の物理クラスタ上に仮想クラスタを立ち上げ，その内部へのシステムインストール時間を計測する．仮想クラスタを構成する各仮想計算機は，メモリ 640MB およびディスク 20GB (Rocks がインストール時に要求する最小構成) を割り当てて，物理計算機上に各一台ずつ起動する．つくば側の Rocks フロントエンドから仮想クラスタ内部へシステムを導入する場合と，秋葉原側の Rocks フロントエンドから VPN を経由してシステムを導入する場合とを比較する．このとき仮想クラスタのノード数を順次変更して実験を繰り返すほか，仮想クラスタではなく物理クラスタに直接システム導入した場合も参考のために計測した．インストール時間を計測するために，各ノードのディスクの MBR を無効にした上で再起動し，強制的に PXE ブートによる再インストールを行っている．インストールされるシステムは Condor や Sun Grid Engine などを含んだもので，転送されるパッケージサイズの合計は 800MB 程度であり，構築後には 2.9GB 程度のディスク容量を占める．

実験結果を**図 6** に示す．秋葉原の Rocks フロントエンドから VPN を経由してインストールする場合と，つくばのローカルネットワークに存在する Rocks フロントエンドからインストールする場合を比較 (図中 b と a，および d と c 参照) すると，VPN 経由の場合においては 200 秒程度所要時間が増加している．所要時間が約 20%増えたものの，問題なくインストールが完了し仮想クラスタを使用することができる．また，同時にインストールを開始するノード数を増やすと，それに応じて各ノードのインストール時間も増加する．仮想クラスタと物理クラスタとの比較 (図中 a と c，および b と d) においては，仮想クラスタの方がインストール時間が 100 秒前後長くなっている．仮想計算機内部の OS への往復遅延は秋葉原の Rocks フロントエンドから 4.7ms 程度また可用帯域は 86Mbps 程度であり，仮想マシンモニタによる若干のオーバヘッドがネットワーク通信において存在するからである．

4.2 仮想クラスタ構築における VPN 通信量

図 7 において，OpenVPN ノードにおいて計測した VPN 上を流れる通信量を時系列に示す．秋葉原の Rocks フロントエンドからつくばの仮想計算機 6 台に対して同時にインストールを開始した時刻から，すべてのノードでインストールを完了する時刻までを描いている．すべてのノードではほぼ同時にインストールが進行しており，BIOS から起動すると PXE ブートによりカーネル (vmlinuz) および初期システムイメージ (initrd.img) を tftp にて取得している．その後 6600 秒付近から Rocks インストーラのシステムイメージ (updates.img, product.img および stage2.img などお

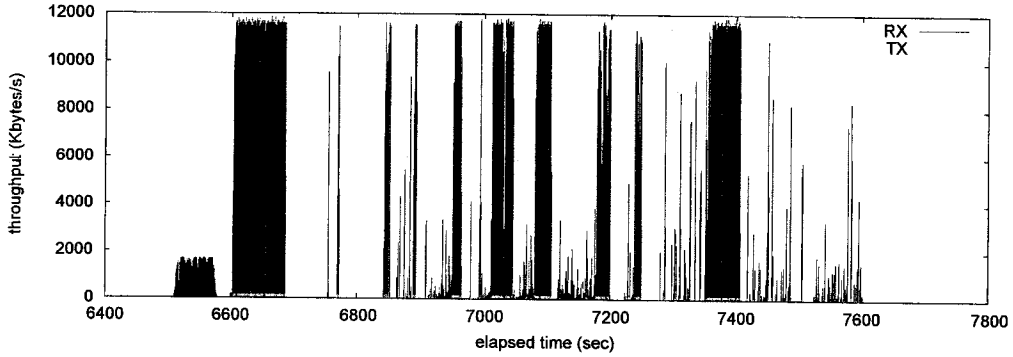

図 7 仮想クラスタ構築時の VPN 通信量 (秋葉原の Rocks フロントエンドから, つくばの仮想計算機 6 台に対して同時にインストールを開始した場合を示す. 通信量の大半を秋葉原からつくばへの通信 (RX) が占める. 図中においてつくばから秋葉原への通信 (TX) はおおむね 250KB/s 以下である.)

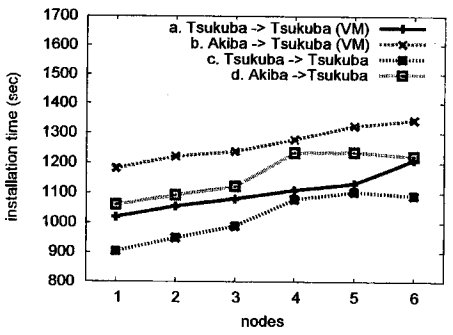

図 6 インストール時間の計測結果

およそ 150MB 程度) を http にて転送している. 6700 秒付近で, anaconda インストーラを立ち上げ, ハードディスクにパーティションを切り, ファイルシステムを作成している. 以降 RPM パッケージの取得と展開および設定を繰り返し, 7600 秒付近でブートローダのインストールなど最終設定を開始している. この図からわかるように, 仮想クラスタ構築時間全体のうちデータ転送に要する時間はあまり大きくない. しかし, ノード数が増加したり可用帯域が小さかったりすると相対的にデータ転送期間の占める割合が大きくなり, クラスタ構築の所要時間が増加してしまう. 複数サイト間で仮想クラスタを構築する際には問題となりうる可能性がある.

4.3 bittorrent 配信機能

Rocks においては, クラスタ構築時にフロントエンドからインストール先ノードへの通信量を削減するため, bittorrent による RPM パッケージ配信機能を備えている. インストール対象のノード群において, すでに RPM パッケージを取得したノードが存在する場合, 他のノードはそれらのノードからも RPM パッケージの取得を試みる. bittorrent 機能は VPN を介して仮想クラスタを構築する際にも同様に働くことが期待され, VPN 通信によるボトルネックを緩和できる可能性がある. 実際, この bittorrent 機能は本実験中もあらかじめ有効に設定されており, ノード間で一部の RPM パッケージが転送されていることが確認された. しかしながら, bittorrent 機能を無効にして再度インストール時間を計測したが有意な差は見られなかった. 今回の実験環境のようにノード数が比較的小さく VPN 帯域が比較的大きい場合においては, Rocks フロントエンドの通信量削減による効果は目に見える形で現れなかったのではないかと推測される. また RPM パッケージは完全に取得完了後に bittorrent にて提供されることから, 各ノードでインストールがほぼ同時に進行している場合には bittorrent はあまり用いられないと考えられる.

4.4 VPN 帯域の制限

VPN の通信帯域を 10Mbps に制限して仮想クラスタの構築を試みた☆. しかし, 各ノードにおいて大きなサイズの RPM パッケージの転送に失敗しインストールが完了しなかった. これは Rocks インストーラにおける RPM パッケージ取得のタイムアウト値 (60 秒) が RPM パッケージサイズ (eclipse の 100MB) に対して十分でないことが原因である. タイムアウト値をあらかじめ十分な値 (この場合 80+α 秒以上) に設定する必要がある. 我々の仮想クラスタ管理システムで用いている Rocks はあらかじめ LAN を想定して開発されている. そのため, VPN をまたいで仮想クラスタを構築する際には, Rocks 自体への若干の修正は避けられないといえる. なお, 一度インストールさえ完了

☆ OpenVPN には shaper オプションが存在し帯域制限が可能であるものの, クライアント/サーバモードとして動作する際には利用できない. かわりに `cbq.init` を用いた.

すれば, その後帯域を 10Mbps に制限してもクラスタ管理システム自体には目立った問題は出なかった.

5. まとめ

複数サイトに対して横断的に仮想クラスタを構築できれば, アプリケーションごとに最適化された単一の実行環境をより大規模かつ柔軟に構築できる. このとき仮想クラスタ管理システムは, 単一実行環境としての透過性, 導入および管理の容易さ, および仮想計算機の再配置を実現する必要があると考えている.

本論文では, 複数サイトからなる仮想クラスタを構築するために, イーサネット VPN 機構を仮想クラスタ管理システムに導入することを提案した. サイトごとに散在するひとつの仮想クラスタを構成する仮想計算機群に対して, 単一のネットワークセグメントを提供する. 予備的な評価実験においては, つくばおよび秋葉原のあいだで OpenVPN によって確立された VPN を経由して, 仮想クラスタの構築が可能であることを確認した. 仮想クラスタの構築時間が多少長くなるものの, 仮想クラスタ管理システム自体の動作に問題はない. しかし, 仮想クラスタのノード数が多くなった場合や十分な帯域のネットワークが確保されなかった場合には, 仮想クラスタ構築に失敗したり構築時間が増大することが予想される. インストーラのタイムアウト値を調整したり, サイトごとのパッケージキャッシュ機構を設けるなど, 仮想クラスタ管理システムをより最適化することが望ましい.

今後の課題を以下に述べる. 非常に大規模な仮想クラスタを構築した際には, 仮想クラスタ管理の通信のみでネットワークトラフィックが飽和する恐れがある. また, サイト間のネットワーク遅延や帯域の制限は仮想クラスタ内のネットワークにおいて不可避であり, 低遅延あるいは大容量の通信を要求するアプリケーションにとっては問題が生じうる. そこで仮想クラスタ管理用ネットワークセグメントのほかに, 各サイトごとのサブネットからなるアプリケーション用ネットワークも提供することも検討したい. 計算機資源やネットワーク資源の予約機構との連携も期待される.

さらに, クラスタ管理システムにおいて Xen による仮想化が可能になれば, サイト間をまたいだ仮想計算機の再配置に取り組む予定である. また, サイト間をまたいだストレージアクセスにおいては, I/O 性能の劣化を無視できない可能性があり, ストレージノードの再配置なども視野に入れる. 基本的にサイト間の VPN 接続は仮想クラスタ提供中は持続的に提供されなければならない. そこで VPN 接続の多重化や経路の冗長化も今後調査する.

謝辞 本研究に際して貴重なご意見を頂いた仮想クラスタ管理システムに関わる諸氏に感謝する.

参考文献

1) Barham, P., Dragovic, B., Fraser, K., Hand, S., Harris, T., Ho, A., Neugebauer, R., Pratt, I. and Warfield, A.: Xen and the art of virtualization, *Proceedings of the nineteenth ACM symposium on Operating systems principles*, New York, NY, USA, ACM Press, pp. 164–177 (2003).
2) Housley, R. and Hollenbeck, S.: EtherIP: Tunneling Ethernet Frames in IP Datagrams, RFC 3378 (2002).
3) LAN MAN Standards Committee of the IEEE Computer Society: *IEEE standards for local and metropolitan area networks: virtualbridged local area networks*, IEEE (1999).
4) Lau, J., Townsley, W.M. and Goyret, I.: Layer Two Tunneling Protocol - Version 3 (L2TPv3), RFC 3931 (Proposed Standard) (2005).
5) OpenVPN: `http://openvpn.net/`.
6) PacketiX VPN: `http://www.softether.com/`.
7) Papadopoulos, P. M., Katz, M. J. and Bruno, G.: NPACI Rocks: Tools and Techniques for Easily Deploying Manageable Linux Clusters, *Cluster 2001: IEEE International Conference on Cluster Computing* (2001).
8) Satran, J., Meth, K., Sapuntzakis, C., Chadalapaka, M. and Zeidner, E.: Internet Small Computer Systems Interface (iSCSI), RFC 3720 (2004).
9) VMware Infrastructure: `http://www.vmware.com/`.
10) Vtun: Virtual Tunnels over TCP/IP networks: `http://vtun.sourceforge.net/`.
11) 中田秀基, 横井威, 江原忠士, 谷村勇輔, 小川宏高, 関口智嗣: 仮想クラスタ管理システムの設計と実装, 第 5 回先進的計算基盤システムシンポジウム SACSIS 2007 (2007).