情報処理学会論文誌　コンピューティングシステム　Vol. 4　No. 2　12–24 (Mar. 2011)

アプリケーション層プロトコルの状態を考慮した広域ネットワーク上での仮想マシン移送

石川　豊[†1]　山田浩史[†1,†2]　浅原理人[†1]
花岡美幸[†1]　河野健二[†1,†2]

広域ネットワーク（WAN）上での仮想マシン移送は複数データセンタ間で負荷状況を調整したり，時間帯に応じて電力コストの安い地域に再配置を行ったりする際に有用である．しかし，仮想マシン移送の実行は，仮想マシンを停止させる必要があるためサービスの中断時間が生じてしまい，移送対象の仮想マシン上のサーバに接続しているクライアントに不快感を与えてしまう．特に WAN 上で仮想マシンの移送を行う場合，使用できるネットワーク帯域が狭いことからサービスの中断時間が長くなる．クライアントの不快感はサービスのユーザ数減少を招いたり，他にも，管理者が仮想マシン移送を使用しなくなったりすることにつながり，データセンタ間での柔軟な資源管理の妨げとなる．本論文では仮想マシン移送にともなうダウンタイムをクライアントから感知されにくくする手法を提案する．提案手法はクライアントと仮想マシン上のサーバ間とでやりとりされるアプリケーション層プロトコルに着目する．各コネクションでやりとりされるメッセージを監視し，コネクションの状態を追跡して，コネクションの状態がクライアントにサービスの中断を感知されにくい状態となったときに仮想マシンの移送を実施する．提案手法のプロトタイプを Xen 3.3.0 上に実装し，Web サーバを動作させた仮想マシンを提案手法と従来手法を用いて移送する実験を行った．SPECweb を用いて実験を行ったところ，既存手法を用いて仮想マシンを移送した場合には許容できる時間内に Web サーバが返した応答の割合が全応当の 65.0%に対して，提案手法を用いた場合には 81.0%となった．

Live Wide-area Migration of Virtual Machines by Exploiting Application Layer Protocol Context

Yutaka Ishikawa,[†1] Hiroshi Yamada,[†1,†2]
Masato Asahara,[†1] Miyuki Hanaoka[†1]
and Kenji Kono[†1,†2]

Virtual machine (VM) migration is an effective way to balance loads between data centers, and save electricity costs by migrating VMs to a region where electricity costs are cheaper. Unfortunately, VM migration causes service downtime because the VM must be stopped during its migration. This service downtime becomes strongly pronounced in VM migration across Wide Area Network (WAN) since the bandwidth of WAN tends to be narrower than that of Local Area Network. The downtime of VM migration disables clients to regularly enjoy the service, leading to the decrease of the service popularity. It also prevents administrators from conducting VM migration, which obstructs flexible resource management across data centers. This paper presents a new technique that makes less clients aware of service downtime stemming from VM migration. We exploit the context of application layer protocols between clients and a server. Our technique traces the state of connections between clients and a server running on the VM, and conducts VM migration when the state of the connections becomes a state in which less clients are aware of the service downtime. We implemented our prototype on Xen 3.3.0 and conducted experiments with SPECweb benchmark. The experimental results demonstrate that our technique improves the response performance of the Web server. The responses with our technique within the tolerable time were 81.0% of all the responses. On the other hand, the responses with existing technique were 65.0%.

1. はじめに

データセンタにおいて仮想化技術[5] を用いたサーバ資源の統合が広く導入されるようになった．サーバ資源の統合を行うと，CPU やメモリ，ネットワーク帯域といった資源を物理マシン上の仮想マシン間で共有し，必要に応じて各仮想マシンに割り当てることで資源を効率良く使うことができる．AFCOM の調査によると，2010 年現在でデータセンタの約 72.9%が仮想化技術を導入している[1]．

ワイドエリアネットクワーク（WAN）で接続された複数のデータセンタ間において，仮想マシン移送を用いることで柔軟な資源管理が行える．仮想マシン移送とは，仮想マシンのメモリ内容や CPU レジスタ，仮想ディスクの内容をコピーすることで，仮想マシンを実行したまま別の物理マシン上に移動させる操作のことである．この操作 1 つで別の物理マシンへ実行環境を移動できるため，従来のように，移動元の仮想マシンの停止や移動先と移動元の仮想マシン内のファイルの一貫性の維持，移動先の仮想マシンの起動といった操作は不要となる．仮想マシンの移送には WAN 上での移送とローカルエリアネットワーク（LAN）

†1 慶應義塾大学理工学部情報工学科
Department of Information and Computer Science, Faculty of Science Technology, Keio University
†2 科学技術振興機構戦略的創造研究推進事業
Core Research for Evolutional Science and Technology, Japan Science and Technology Agency

上での移送が存在するが，特に WAN 上で仮想マシンの移送を行うことで仮想マシンの配置の自由度が向上し，さまざまな恩恵を受けることができる．たとえば，あるデータセンタ内の計算機資源が枯渇したとき，仮想マシン移送を行うことで資源を占有している仮想マシンを別のデータセンタに移動させることができ，データセンタ内の資源の枯渇を防ぐことができる．また，仮想マシン上でサーバを動作させている物理マシンをメンテナンスのために停止させるとき，LAN 上に代替サーバを用意する余裕がない場合でも WAN 上での移送を行うことで仮想マシンサーバを別のネットワーク上に移動させることができ，メンテナンス中もサービスを提供しつづけることができる．さらに，WAN 上での移送を用いると，電力価格が安い時間帯の地域に仮想マシンを移動させて電力コストを抑えるといったことが可能になる．

しかしながら，WAN 上での仮想マシン移送はダウンタイムが長くなりがちであり，サービスの長期停止によって多くのクライアントに不快感を与えてしまう．仮想マシンの移送を行う際には，移送先と移送元とで CPU レジスタやメモリ，仮想ディスクの一貫性を保つために，仮想マシンを 1 度停止しなければならない．仮想マシンを停止すると仮想マシン上で動作しているサーバも停止するので，クライアントへ提供しているサービスの中断が起こる．WAN はさまざまなトラフィックの流れる複数のネットワークで構成されており，移送開始時に十分な帯域が使用できたとしても，経由するネットワークの使用状況の変化するため，移送途中に帯域が急激に狭くなることもある．そのため，WAN 上での仮想マシンの計算機資源の転送時間が長くなりがちである[3]．これにより，仮想マシン移送によるサービスの停止は避けることができず，移送を行うと，移送対象の仮想マシン上のサーバに接続しているクライアントに不快感を与えてしまう．クライアントの不快感は，最悪の場合，サービスのユーザ数減少につながる可能性がある．また，管理者はクライアントが不快感を感じることを恐れて仮想マシン移送を行わなくなり，データセンタ間の柔軟な資源管理ができなくなってしまう．

本論文では仮想マシン移送のダウンタイムをクライアントから感知されにくくする手法を提案する．提案手法ではサーバとクライアント間でやりとりされるアプリケーション層プロトコルに着目する．提案手法は各コネクションでやりとりされるメッセージを監視し，コネクションの状態を追跡する．コネクションの状態がクライアントにサービスの中断を感知されにくい状態であるときを検出し仮想マシン移送を実施する．サービスの中断をクライアントに感知されにくいコネクションの状態には，以下の 2 種類の状態が考えられる．1 つは，サーバがクライアントにサービスを提供していない状態である．サーバとクライアントの間でコネクションが確立していてもサービスに関するデータのやりとりをしていない場合は，コネクションが中断されてもクライアントは影響を受けない．そのため，移送によりサービスの中断が発生してもクライアントはそのことに気付きにくい．もう 1 つは，十分に大きいデータが転送されている状態である．転送データが大きければ転送完了までの総時間は長くなる．この時間が移送による中断時間よりも十分長ければ，相対的に移送による中断時間は無視できるほど小さくなり，クライアントはサービスの中断が発生したことを感知しにくくなる．

提案手法には次の特徴がある．提案手法は WAN 上において仮想マシン移送によるサービスの中断時間をクライアントに感知されにくくする．これにより仮想マシン移送のダウンタイムが与えるクライアントの不快感をできるだけ小さくする．また，提案手法は仮想マシン移送のダウンタイムを感知されにくくすることで，管理者が仮想マシン移送を実施するための管理，計画コストを削減する．これにより，積極的なデータセンタ間の仮想マシン再配置による負荷分散や省電力を促す．

ここで，本論文で示す方式は文献 3) などで提案されている WAN 上での仮想マシン移送技術に組み込んで利用することを想定している．通常，WAN 上での仮想マシン移送は仮想マシンの IP アドレスの変更をともなう．文献 3) の方式では，IP トンネリングと Dynamic DNS を併用することで，仮想マシンの IP アドレス変更にともなうクライアントのコネクション切断といった副作用を最小限に抑えている．本論文で示す方式もそれと同様の方式でクライアントとのコネクションを維持することが可能である．

提案手法のプロトタイプを Xen 3.3.0[2] 上に実装し，提案手法の有効性を示すために実験を行った．実験では，仮想マシン上の Web サーバに対してベンチマーク SPECweb2005[7] を用いて負荷をかけ，WAN 環境を再現したネットワーク上で仮想マシンの移送をベンチマークの動作中に 1 回行った．6 回の実験を行った結果，提案手法を用いた場合は平均して 81.0%の応答が SPECweb が許容できると定義する時間内に返ったのに対して，既存手法を用いた場合は平均して 65.0%の応答が SPECweb が許容できると定義する時間内に返った．よって，提案手法を用いた方が，仮想マシンの移送によって発生するサービスの中断をクライアントに感知されにくいということが示された．

2. 関 連 研 究

これまで，CPU やメモリに着目して仮想マシン移送にともなう停止時間を削減する手法が提案されている．Live Migration[4] は，移送を行う前にあらかじめメモリイメージを転送

しておくことで，仮想マシンを移送する際に転送するデータの総量を小さくし，仮想マシンの停止時間を短縮する移送方法である．Live Migration では，仮想マシンを動作させたままの状態ですべてのメモリページを移送先の物理マシン上に転送する．その後，更新があったページの再送を繰り返し，更新ページが十分に少なくなったら仮想マシンを停止させて，残りのメモリページと CPU レジスタ値を移動させる．Post-Copy Based Live Migration [6] は，先に CPU の内容を転送した後にメモリページを徐々に転送する仮想マシン移送の停止時間削減手法である．メモリの転送が終わっていない状態で移送先で仮想マシンを再開し，アクセスしたメモリページがまた転送されていない場合には移送元の物理マシンからネットワークを通してページを取得する．これらの手法は LAN 環境において仮想マシンはすべてネットワークストレージを共有している環境での移送を想定しており，移送時に仮想マシンのディスクイメージの転送を想定していない．たとえば，複数データセンタを利用した代表的なサービスである Amazon EC2 では，各仮想マシンはそれぞれ仮想ディスクを備えており，こうした環境では利用することができない．

Live Wide-Area Migration [3] は Live Migration の移送プロセス中にディスクイメージの転送も行うことで，WAN 上での仮想マシンの移送を行えるようにした手法である．論文では，Live Wide-Area Migration を用いて WAN 上での仮想マシンの移送を行った場合，サービスの中断時間は約 67 秒になると報告されている．この手法ではコネクションの状態を考慮せずに移送を行うので，サービスの中断がクライアントに感知されやすい．

Sandpiper [8] は物理マシンの資源の使用状況を考慮し，仮想マシンに資源が行きわたるよう適切に仮想マシンを移送するシステムである．Sandpiper は各仮想マシンの CPU，ネットワーク，メモリの使用率を考慮し，複数の物理マシンで構成されたデータセンタのような環境の上で自動的に仮想マシンの再配置を行う．Sandpiper は各物理マシンの資源の使用状況を考慮するが，コネクションの状態を考慮して移送のタイミングを探すことはしない．そのため，移送の際にサービスの中断をクライアントに感知されやすい．

ここで，仮想マシン移送と既存のレプリケーションを用いたサーバ移動とを比較する．両者はどのようなサーバを移動させたいかにより使い分ける必要がある．レプリケーションを用いてサーバを移動させる場合，移送元と移送先とでセッションといったサーバが保持する状態の一貫性を維持するコストが発生するが，ダウンタイムは仮想マシン移送を用いた場合と比べると短くなる．一方，仮想マシン移送を用いる場合，ダウンタイムはレプリケーションを用いた場合と比べると長くなるが，サーバが保持する状態を管理するコストは低くなる．たとえば，状態を持たないステートレスな Web サーバを遠隔ホストに移動させたい場合には，状態を維持する必要がないのでレプリケーションが適している．また，セッションのような状態を持つステートフルなサーバを移動させる場合には，仮想マシン移送が適しているといえる．

3. 提　　案

3.1 概　　要

本論文は，仮想マシン移送によるダウンタイムをクライアントに感知されにくくする手法を提案する．仮想マシン移送のダウンタイムを感知されにくくするために，本手法ではサーバとクライアント間でやりとりされているメッセージの内容を解析し，各コネクションの状態を追跡する．コネクションがサービスの中断時間をクライアントに感知されにくい状態であれば，そのときに仮想マシンを移送することで仮想マシン移送のダウンタイムをより多くのクライアントに気付かせないようにする．

ダウンタイムを感知しにくいコネクションの状態として次の 2 種類の状態がある（**図 1**）．1 つめはコネクションの状態が待機状態であるときである．待機状態とは TCP コネクションは確立しているが実際にメッセージのやりとりを行っていない状態のことである．待機状態中にサーバが停止しても，クライアントはサービスの提供を受けていないため，サーバの停止に気付くことは難しい．待機状態の例として HTTP の Keep-Alive 状態があげられる．Keep-Alive 状態とは，HTTP においてリクエストに対応するレスポンスの処理が終わった後，次のリクエストを受信するまでの状態である．

移送に適したもう 1 つの状態は，転送中のデータのサイズが十分大きい場合である．転送中のデータのサイズが大きいと，移送による中断時間がデータの総転送時間と比べて相対的に小さくなるため，クライアントはサービスの中断による影響を感知しにくい．HTTP の GET 要求の応答で大きなファイルを転送している場合などがこれにあたる．

仮想マシンを移送する際には仮想マシン間の依存性を考慮しなければならない．移送後にサーバがサービスを正しく提供できなくなる可能性があるからである．そのため，管理者は移送後もサービスが引き続き継続できるよう仮想マシンを移送する必要がある．たとえば，あるサーバが他の仮想マシンで稼働しているアプリケーションサーバやデータベースサーバといったバックエンドサーバと連携している場合を考える．これらの仮想マシン群を移送する場合，アプリケーションサーバとデータベースサーバとのメッセージのやりとりも考慮して移送をしないと，移送のために仮想マシンが停止する瞬間に，あるリクエストの処理結果を仮想マシン上のサーバが取りこぼす可能性がある．そのため，移送後もクライアントに一

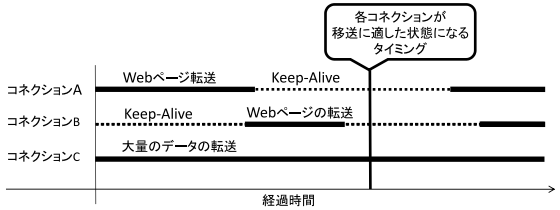


図 1　コネクションの状態遷移と移送タイミング
Fig. 1　Connection states and migration timing.

貫したサービスを提供できるように，バックエンドの依存性を解決する機構を別途用意する必要がある．

3.2　アプリケーション層プロトコルを用いたコネクションの状態の解析

提案手法では，サーバが確立しているコネクションの状態を知るために，サーバ・クライアント間でやりとりされるメッセージをアプリケーション層プロトコルに基づいて解析する．アプリケーション層プトロコルとはアプリケーション層で行われる通信の規約のことであり，HTTP や POP，SMTP がその代表例である．アプリケーション層プロトコルは複数の状態を持ち，メッセージをやりとりすることで状態遷移していく．これにより，コネクション上でやりとりされるメッセージを解釈して追跡することで，コネクションが現在どの状態にあるかを知ることができる．

あるコネクションが待機状態であることを判断するために，対象のアプリケーション層プロトコルのどの状態が待機状態に分類されるかをあらかじめ定義しておく．サーバとクライアント間でコネクションが確立した際には，プロトコルの定義に従ってメッセージを解釈してコネクションの状態を追い，コネクションの状態が待機状態に分類した状態になったかを判断する．

また，サイズの大きいファイルを転送しているコネクションを検出するために，アプリケーション層プロトコルのメッセージから転送するファイルサイズを取得する．アプリケーション層プロトコルメッセージからは，転送対象のデータ以外に複数の情報を入手することができる．たとえば，一部のアプリケーション層プロトコルはデータを転送する前にデータサイズを相手に通知する．提案手法はこれを利用して，データの転送が始まる前にデータサイズを取得することで，コネクションが大量のデータを取り扱うか否かを判断する．

HTTP を例にして，アプリケーション層プロトコルから移送に適した状態をどのように検出するかを説明する．HTTP の状態遷移を簡単に表すと**図 2** のようになる．クライアントが HTTP サーバに接続をし，リクエストを送信してサーバからデータを取得したあと，リクエスト待ち状態になる．このリクエスト待ちの状態は待機状態であるので移送に適したタイミングと定義する．リクエスト待ち以外の状態で仮想マシンの移送が行われると，レスポンスの転送完了までの時間が仮想マシンの停止時間分だけ延長されるのでサービスの中断を感知されやすくなる．よってデータ取得後のリクエスト待ち以外の状態は，仮想マシンの移送に適さない状態とする．ただし，データ転送状態で大量のデータを転送するならば，仮想マシンの停止時間はレスポンスの転送完了までの時間と比較して十分に小さくなると考えられる．その場合は，データ転送状態も仮想マシンの移送に適した状態と定義する．データ転送状態での転送量を把握するために HTTP の場合はヘッダフィールド内の Content-Length の値を参照する．

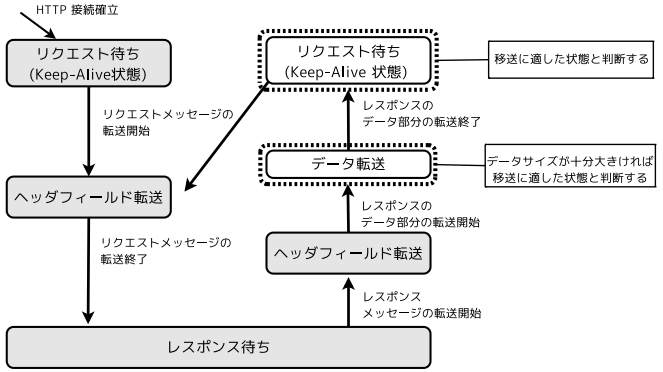


図 2　簡単な HTTP の状態遷移図：簡単のためリクエストメッセージはヘッダフィールドしか持たないものとする
Fig. 2　Simple HTTP state transition.

ここで，リクエスト待ち状態に至る経緯として 2 種類考えられる．1 種類目はクライアントがページを取得した後に遷移する場合である．2 種類目は HTTP 接続確立直後に遷移する場合である．前者は，クライアントがページを取得してコンテンツを閲覧しているときのリクエスト待ち状態が移送に適している．しかし後者のリクエスト待ち状態では，ページを取得するため，クライアントは多くの場合すぐに HTTP リクエストをサーバに発行すると考えられる．そのため，移送に適しているとはいい難い．そのため，HTTP プロトコル上では同じリクエスト待ち状態であっても，別の状態を設けている．

HTTP と同様に POP3 や FTP でもアプリケーション層プロトコルを追跡して移送に適し

た状態を判断することができる．POP3 の状態はユーザの認証を行う AUTHORIZATION 状態，コマンドを指定する TRANSACTION 状態，コマンドを実行する UPDATE 状態の順に遷移していく．POP3 では TRANSACTION 状態で LIST コマンドが実行されていれば転送対象のメールの合計サイズを POP3 メッセージを解釈することで確認できる．メールのサイズが十分大きければ UPDATE 状態に長い時間がかかると考えられるので，UPDATE 状態を仮想マシンの移送に適した状態とする．それ以外の状態は仮想マシンの移送に適さない状態とする．FTP のコントロール接続は HTTP と同様にコネクションを確立しているだけの待機状態が存在する．この待機状態では仮想マシンの移送を許可する．また，コントロール接続上の FTP コマンドのやりとりから，データ接続で転送されるデータが十分大きいと判断できる場合，そのデータ転送を行っているコネクションを移送に適した状態のコネクションであると判断する．提案手法を用いる場合，大きいサイズのデータを扱うプロトコルの方が仮想マシンの移送に適したタイミングが多く存在することになる．よって，メールのように比較的小さなデータを扱う POP3 プロトコルを用いるサーバより HTTP，FTP のように大量のデータの転送を行うことの多いプロトコルを用いるサーバの方が提案手法を導入した際に効果が現れやすいと考えられる．

3.3　クライアントとのコネクション

WAN 上で仮想マシン移送を行うときには，クライアントのネットワーク接続を考慮しなければならない．WAN 上で仮想マシンを移送すると，地理的な位置が変わるため仮想マシン自身の IP アドレスが変わるからである．そのため，クライアントとのネットワーク接続を考慮せずに仮想マシンを移送すると，クライアントのネットワーク接続は切れてしまい，サービスを継続して受けられなくなってしまう．また，新たにサービスに接続するクライアントは，移送後の新しい IP アドレスを指定してネットワーク接続しなければならなくなってしまう．

本論文の方式は文献 3) のような既存の WAN 上での仮想マシン移送技術に組み込んで使用することを想定している．既存手法はクライアントのネットワーク接続を考慮しながら WAN 上での仮想マシン移送が可能である．たとえば，文献 3) では IP トンネリングと Dynamic DNS とを併用することで，クライアントとのネットワーク接続を考慮して仮想マシンを移送する．クライアントのネットワーク接続を維持するために，仮想マシン移送によって仮想マシンが停止する直前に，移送元と移送先との間で IP トンネルを作成する．作成した IP トンネルによって，いままで接続していたコネクションからのパケットは移送先に転送される．また，新たにホスト名を検索するクライアントのために，仮想マシン移送が完了した直後，Dynamic DNS のエントリを更新する．こうすることで，クライアントは移送前と同様の方法でサービスを受けることができる．また，仮想マシン移送によって仮想マシンが停止している間はサービスを提供することができないため，この間は到着するパケットをドロップしている．

4. 実　　装

提案手法のプロトタイプを Xen 3.3.0 に実装した．OS には Xen 用に修正を加えられた Linux を使用した．

実装は Xen の特権仮想マシンである Domain-0（Dom-0）上に行った．Xen では，Xen 上で稼働するゲスト仮想マシンである Domain-U（Dom-U）の通信パケットは特権仮想マシンである Dom-0 を通過する．そのため，すべての Dom-U のコネクションの通信内容を取得するために，Dom-0 上で通信パケットの情報を取得し，各 Dom-U に接続されているコネクションの状態を一括で管理する．

本実装は通信パケット取得部，メッセージ解釈部，仮想マシンの移送実行部に分かれている．各実装部分の関係を**図 3** に示す．通信パケット取得部は Dom-0 上を通過する Dom-U の通信パケットの情報を取得してメッセージ解釈部に渡す．メッセージ解釈部は，まず，通信パケットの情報をもとにパケットの内容を読み出し，コネクションごとに分類してアプリケーション層プロトコルメッセージを復元する．次に，プロトコル解析ルールを用いて復元

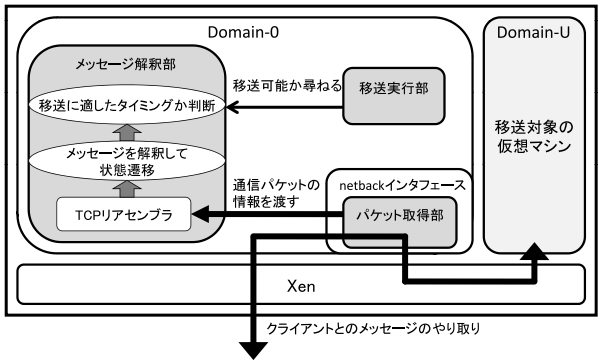


図 3　提案手法の実装の概略図
Fig. 3　Overview.

したメッセージを解析して，コネクションのアプリケーション層プロトコルの状態とコネクションで転送しているデータのサイズを入手する．そして，プロトコルの状態と転送データサイズから，各コネクションが移送タイミングとして適している状態かを判断して，移送タイミングとして適していないコネクションの総数を保存する．仮想マシン移送実行部はアプリケーション層プロトコルメッセージ解釈部が移送タイミングとして適していないと判断したコネクションの総数を参照して，仮想マシンの移送を行うか延期するかを決定する．

アプリケーション層プロトコルを解釈する機構を実装するレイヤとして次の 2 種類の形態が考えられる．1 つ目は Dom-U 上に，2 つ目は Dom-0 上に実装する形態である．各 Dom-U 上でターゲットとなるサーバプログラムとアプリケーション層メッセージ解釈機構とを連動させることで，暗号化といった処理をメッセージに施されても，正しくアプリケーション層プロトコルを追跡することができる．しかし，各 Dom-U ごとにアプリケーション層メッセージ解釈機構を用意する必要がある．また，仮想マシン移送実行部分は Dom-0 上に実装されているため，解釈機構と Dom-0 とが適宜情報をやりとりしなければならず，これらの通信部分を実装しなければならない．

一方，Dom-0 上では暗号処理が施されるとメッセージは解釈できないため，アプリケーション層プロトコルを追跡することが難しい．しかし，メッセージにそのような処理が施されていなければ，各 Dom-U に修正することなくアプリケーション層メッセージを追跡することができる．また，解釈機構が動作するレイヤが仮想マシン移送実行部分と同じになるため，実装が単純になる．本研究では，アプリケーション層プロトコルを利用することで移送のダウンタイムをクライアントから隠蔽可能か否かを示すために，実装が単純な後者の実装法を採用した．

4.1　通信パケット取得部

通信パケット取得部は，Dom-U とクライアントの間でやりとりされる通信パケットの情報を取得してメッセージ解釈部に渡す役割を持つ．Dom-U とクライアント間でやりとりされるメッセージを取得するために，通信パケット取得部は Dom-U 側の Dom-0 のインタフェースである `netback` インタフェース内で稼働する．通信パケット取得部は，`netback` インタフェースを通過するパケットの情報を持つ `socket kernel buffer` 構造体を取得しメッセージ解釈部に渡す．

4.2　メッセージ解釈部

メッセージ解釈部は，アプリケーション層プロトコルの定義を読み込んで，通信パケットを解釈し，各コネクションの状態遷移を追跡し，移送に適さない状態のコネクション数を集計する．アプリケーション層プロトコルの定義は，文献 9) で提案されているプロトコル記述言語を用いて記述するようにした．この言語を用いると，構文解析解析器生成系 yacc のように，プロトコルの状態遷移を宣言的に記述することができる．さらに，yacc におけるアクションと同様に，各状態における処理を C 言語のプログラムとして記述することができる．対象とするアプリケーション層プロトコルごとにその定義を記述すれば，本論文の方式は多くのプロトコルに対して適用可能である．

図 2 に示した簡単化した HTTP の場合，5 つの状態とそれらの状態間での遷移を起こすメッセージのフォーマットを宣言的に記述する．リクエスト待ちの状態にあるときは，移送に適さないコネクション数を 1 つ減らすようにアクションを記述する．また，ヘッダフィールドを転送している状態では，Content-Length の値を取得するアクションを記述しておく．これによって，データ転送中には Content-Length の値が十分に大きければ，移送に適さないコネクション数を 1 つ減らすようなアクションを記述できるようになる．ここの部分の記述は具体的に以下のようになる．

```
1. regexpr num = ('1'-'9')('0'-'9')*
2. state server_header_responce {
3.  :>
4.  "Content-Length:" (num)
5.  [$check_content_length(atoi(num))$]
6.  -> general_request;
7. }
```

Content-Length 値を取得するために，文字列から数字列を抜き取る num を定義する（1 行目）．3，4 行目はサーバのレスポンス内容から Content-Length 値を num を利用して取得する．5 行目の [$ $] 内はアクションを表す．`check_content_length()` は，Content-Length の値を引数にとり，内部であらかじめ定めた閾値と num とを比較して，閾値より小さければ，移送に適さないコネクション数を 1 つ減らす．本実装では閾値を 4 MB としている．num を int 型に変換するために，`atoi()` を利用している．そして次の状態に遷移する（6 行目）．このようにしてアプリケーション層プロトコルごとに移送に適した状態であるか，そうでないかを判別するプロトコル定義を記述することができる．

なお，図 3 に示したように，メッセージ解釈部はパケット取得部から送受信されるパケット列を取得する．これらのパケットは順序どおりに並んでいるとは限らないため，TCP リアセンブラを用いて順序どおりに整列させたうえで，アプリケーション層プロトコルの解釈

を行うようになっている．

4.3　仮想マシンの移送実行部

提案手法は，仮想マシンの移送に Live Wide-Area Migration [3] を使用する．Live Wide-Area Migration は通常の Live Migration の手順を行う前に，仮想マシンのディスクイメージの転送を行う．転送中または転送後に仮想マシンによってディスクイメージに更新が行われた場合には，移送に用いるコネクションとは別のコネクションを通してディスクの更新部分を移送先に転送することでディスクの一貫性を保証する．ディスクイメージの転送後は，通常の Live Migration と同様にメモリページの転送を行う．メモリページの転送を終了する条件は，転送回数があらかじめ指定されている最大回数を超えた場合，転送レートの上限に達し，かつ，今回転送したメモリページ数が前回の転送したメモリページ数よりも大きい場合，今回転送したメモリページ数が 50 を下回った場合，そして，これまでに転送したメモリページ数が一定数を超えた場合である．

本実装では，メモリ転送の終了条件を満たしたときに，アプリケーション層プロトコルメッセージ解釈部が持つ，コネクションの状態の集計結果を参照する．仮想マシンの各コネクションが移送に適した状態であると判断されていれば，クライアントに感知されにくい移送が実行できると判断して，仮想マシンを停止させ，残りのメモリページと CPU レジスタ内容の転送して仮想マシンの移送を行う．一方，移送に適さない状態のコネクションが存在する場合は，再びメモリページの転送を行い，各コネクションが移送に適した状態になるのを待つ．

図 4 は実装したシステムにおける仮想マシンの移送のプロセスを表したものである．図 4 の上側の 3 本の横線はアプリケーション層プロトコルのコネクションの状態遷移を表す．太線の部分はデータの転送を行っている状態を表し，点線の部分は待機状態であることを表す．図 4 の下側の矢印は仮想マシンの移送プロセスを表す．仮想マシンの移送プロセスは，まず，ディスクイメージの転送を行い，続いてメモリページの転送を行う．メモリページの転送が終了条件を満たしたところでコネクションが移送に適した状態であるかを尋ねる．コネクションのいずれかが移送に適さない状態であれば，メモリページの転送を再び行って移送に適したタイミングになるのを待つ．各コネクションが移送に適した状態であれば，仮想マシン移送を実行する．

本プロトタイプではすべてのコネクションが移送に適した状態にならなくても移送を行う．すべてのコネクションが移送に適した状態になるまでに移送を遅延させると，移送を指示してから完了するまでに長い時間がかかってしまう恐れがあるためである．特にリクエストが集中しているような高負荷のサーバの場合，すべてのコネクションが移送に適した状態になる可能性はかなり低く，移送の必要性があるにもかかわらず移送が完了するまでに長い時間がかかることが予想される．本プロトタイプでは，移送に適さないコネクションがあった場合には移送を延期しその延期が長くなりすぎないようにするという移送のリトライ機構を実装した．本リトライ機構は，Ethernet の衝突回避アルゴリズムの 1 つである制限付き二進指数バックオフアルゴリズムを応用し，許容するコネクション数を徐々に増やす．具体的には初期状態では移送に適さないコネクションが 1 つでも存在すると移送を行わないが，4 回移送が延期されると移送に適さないコネクションが 1 つ存在しても移送する．以降，4 回移送が延期されるごとに，2，4，8，16 というように許容する移送に適さない状態のコネクション数が増える．

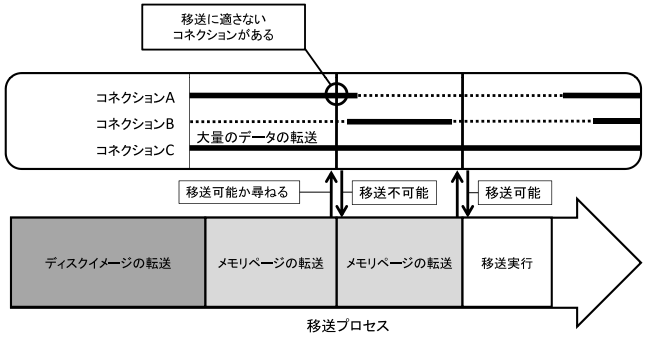


図 4　提案手法を用いた移送プロセス
Fig. 4　Migration process.

ここで，移送のリトライ機構は用途に応じて適宜変更する必要がある．採用する手法によって移送の挙動が変わり，サービスの品質に与える影響が変わるためである．そのため，移送に要する時間の限度やダウンタイム隠蔽の重要度を考えながら，適宜変更しなければならない．たとえば，移送に要する時間はかかってもよいのでクライアントからダウンタイムをできるだけ隠蔽したい場合には，移送に適さないコネクションが 1 つでもある場合には移送を延期するようなリトライ機構を採用する必要がある．また，クライアントにダウンタイムを多少気付かれてもよいので移送を早く実行したい場合には，移送に適さないコネクションが多少あっても移送を実行するようなリトライ機構を採用する必要がある．

5. 実　　験

本実験では，提案手法を用いることで WAN 上での仮想マシンの移送がクライアントに感知されにくくなることを確認する．仮想マシン上の Web サーバにベンチマークで負荷をかけ，提案手法と既存手法を用いて WAN 上での仮想マシンの移送を行い，既存手法を用いた場合のベンチマークのスコアと提案手法を用いた場合のベンチマークのスコアを比較した．また，実験の結果を用いて仮想マシンの移送の総時間，仮想マシンの停止時間，メモリページの転送量，Web サーバの平均応答時間の評価，提案手法のオーバヘッドの評価を行い，提案手法を用いた場合と既存手法を用いた場合で比較した．なお，ベンチマークには Web サーバの標準的なベンチマークである SPECweb2005 [7] を使用した．

5.1 実 験 環 境

実験には 4 コアの Xeon 2.66 GHz CPU，メモリ 2 GB，66 GB のハードディスクで構成された物理マシン 4 台を使用した．これらの物理マシンから，仮想マシンを動作させるマシンに 2 台，SPECweb のクライアントに 1 台，ネットワーク帯域の制御用に 1 台を割り当てた．物理マシン間のネットワーク接続は**図 5** のようになっている．仮想マシンを動作させる 2 台のマシンと SPECweb クライアントマシンは 1 つのスイッチにギガビットイーサネットで接続されている．仮想マシンを動作させるマシンどうしは，帯域制御用のマシンを間に挟んで別のネットワークにつながれている．帯域制御用のマシンは仮想マシンの移送元と移送先のマシンの間に WAN に近いネットワーク環境を作り出す．

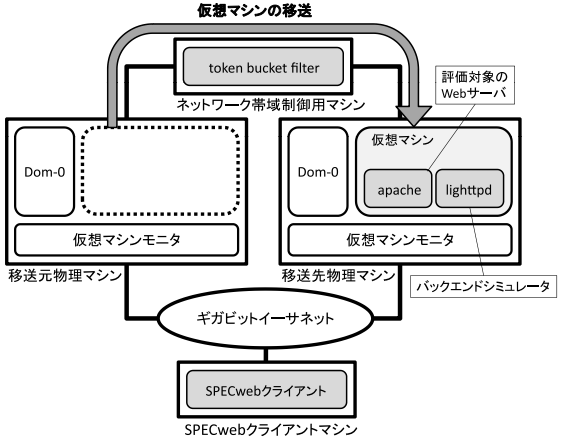


図 5　実験環境のネットワーク
Fig. 5　Experimetal environment.

仮想マシンを動作させる 2 台のマシンには提案手法を実装した Xen 3.3.0 をインストールした．Xen の特権仮想マシンである Dom-0 のカーネルには Linux-2.6.18-xen を使用し，Dom-0 への割当てメモリは 1024 MB とした．移送の対象となる仮想マシンにも Linux-2.6.18-xen カーネルを使用し，割当てメモリは 512 MB，仮想 CPU は 2 個，ディスクイメージのサイズは 8 GB とした．移送される仮想マシン上で動作する Web サーバには Apache 2.2.13 を使用し，SPECweb を動作させるために使用する PHP モジュールには php 5.3.0 モジュールを使用した．SPECweb2005 の実行環境では，通常 Web サーバの後ろで動作しているアプリケーションサーバやデータベースサーバを用意する代わりに，バックエンドサーバのシミュレータを必要とする．バックエンドシミュレータとして移送対象の仮想マシン上で lighttpd 1.4.25 を動作させ FastCGI 2.4.0 モジュールを使用した．評価対象の Web サーバに HTTP リクエストを発行する SPECweb のクライアントを動作させるマシンの OS には Fedora8 を使用した．帯域制御用マシンの OS には Fedora10 を使用した．帯域制御用のマシンの上では token bucket filter が動作している．実験では，この token bucket filter を用いて移送元と移送先のマシン間のネットワーク帯域を任意の帯域に絞り込み，通信遅延 100 ms を付加することでデータベース間をつなぐ広帯域の WAN 環境を再現している．

SPECweb2005 は 3 種類のワークロードを提供しているが，本実験ではその 1 つである Support ワークロードを使用した．Support ワークロードはコンピュータ業者のサポートサイトを模したワークロードであり，さまざまなサイズのファイルを扱う．SPECweb のパラメータには SPECweb が推奨する値を設定した．ただし，仮想マシンの移送がサービスに与える影響を明確にとらえるために，同時接続数（SIMULTANEOUS_SESSIONS）を 260，スコアの計測時間（RUN_SECONDS）を 300 秒，準備時間（WARMUP_SECONDS）を 850 秒に設定した．

5.2 実 験 結 果

5.2.1 Quality of Service の比較

SPECweb のスコアの計測中に仮想マシンの移送を 1 回行った．token bucket filter を用いて移送元と移送先の間の帯域を 100 Mbps，200 Mbps，500 Mbps，1 Gbps に設定し，さらに提案手法を用いた場合と既存手法を用いた場合とで各 6 回実験を行った．本実験で

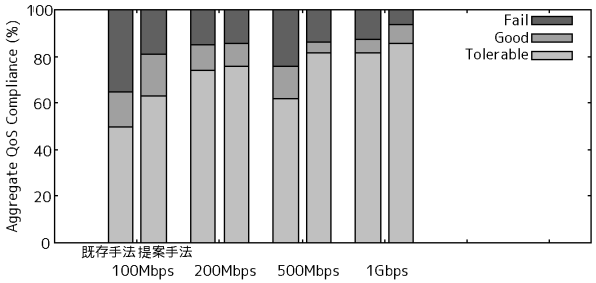


図 6　提案手法と既存手法の QoS のスコア
Fig. 6　SPECweb score.

用いた Support ワークロードには通常のリクエストと比較的大きなファイルの転送を行う download リクエストがあり，QoS（Quality of Service）の評価基準が異なっている．通常のリクエストの場合，応答時間が 3 秒以内ならば十分に早く応答が返ってきているものと見なされ QoS は Good となる．5 秒以内ならば許容できる応答速度であると見なされ QoS は Tolerable となる．5 秒よりも長い時間がかかった場合は応答に失敗したものと判断され QoS は Fail となる．一方，download リクエストの場合はデータの平均転送速度から QoS が評価される．99 KB/sec 以上ならば十分な転送速度であると判断され QoS は Good となる．95 KB/sec 以上ならば許容できる転送速度であると判断され QoS は Tolerable となる．転送速度が 95 KB/sec 以下ならば応答に失敗したものと判断され QoS は Fail となる．

図 6 に実験で得られた提案手法と既存手法の QoS スコアの平均を示す．いずれの帯域のネットワークを使用したときも，提案手法を用いた方が既存手法よりも QoS スコアが向上している．よって，提案手法を用いることで仮想マシンの移送によるサービスへの影響を緩和できていることが示された．特に 100 Mbps の帯域制限をかけた場合，提案手法の Good と Tolerable の合計の割合は 81.0%となるのに対し既存手法の Good と Tolerable の合計の割合は 65.0%となり，QoS スコアが最も向上した．提案手法は特に帯域の狭い WAN 上で移送を行う場合に効果を発揮するといえる．

スコアの値を分析するために，仮想マシンが停止する瞬間の移送に適さないコネクション数を比較した．比較のために，提案手法を用いた場合において，移送に適さないコネクションがメモリ転送終了時にどれだけあるかを記録した．記録したコネクション数は，既存手法を用いた場合において，ダウンタイムの影響を受ける移送に適さないコネクション数であるといえる．これらと提案手法使用時に影響を与えた移送に適さないコネクション数とを比較する．

表 1　移送時に影響を受けた移送に適さないコネクション数
Table 1　Affected connections unsuitable for VM migration.

バンド幅	既存手法				提案手法			
	最大	最小	平均	分散	最大	最小	平均	分散
100 Mbps	42	0	13.6	101.1	1	0	0.2	0.1
200 Mbps	55	0	17.2	115.4	1	0	0.8	0.1
500 Mbps	18	0	9.3	16.3	1	0	0.7	0.2
1 Gbps	22	0	15.2	14.9	1	0	0.8	0.1

提案手法と既存手法とで影響を与えた移送に適さないコネクション数を**表 1** に示す．表 1 は，移送しないコネクション数の最大値，最小値，平均値，分散値を表している．表 1 より提案手法を用いると，移送に適さないコネクション時に移送を行っていないことが分かる．本実験では，既存手法を用いた場合，移送に適さないコネクション数が最大で 55 であった．一方，提案手法を用いると最大でも移送に適さないコネクション数がたかだか 1 のときに移送を行っており，SPECweb のスコアに貢献したと考えられる．

ここで，仮想マシン移送のトラフィックは稼働しているサービスの品質に影響する可能性がある．サーバのネットワークトラフィックと差分ディスクや更新ページの転送トラフィックとが干渉するためである．移送のトラフィックによるサービスへの影響を小さくするために，既存の仮想マシン移送機構には移送時のトラフィックを抑える機構が備わっている．たとえば，Xen が提供する Live Migration 機能では，更新ページを送信する量を抑えることができる[4]．こうした機能を併用することで，サービスの品質に与える影響をできる限り抑えながら仮想マシンを移送できる．

以降の評価では，WAN 上での仮想マシンの移送を想定して 100 Mbps の実験結果について評価を行う．

5.2.2　移送プロセスの総時間の比較

仮想マシンの移送命令が発行されて移送プロセスが開始してから仮想マシンの移送が終了するまでの時間は，可能な限り短いことが望ましい．提案手法を用いた場合，コネクションの状態を考慮して移送の延期を行うことがあるために既存手法に比べて移送が終了するまでの時間は必然的に長くなる．提案手法が既存手法に比べてどの程度移送の実行を遅らせるかを確認し，移送の迅速さが保たれているかを評価した．本評価では，提案手法と既存手法で移送プロセスが開始してから移送が終了するまでの時間を比較する．

移送経路の帯域を 100 Mbps に設定した場合，移送プロセスの開始から終了までにかかっ

表 2　転送メモリページ数と停止時間の比較
Table 2　Transfered memory pages and stop time.

	平均転送メモリページ数	平均停止時間（sec）
提案手法	48,197	17.61
既存手法	46,762	17.20

た総時間は提案手法を用いた場合平均 927 秒であり，既存手法を用いた場合は平均 889 秒であった．よって，提案手法を用いることによって増加した時間は平均して 38 秒程度であり，既存手法の移送プロセスの総時間の約 4.25%であった．

5.2.3　仮想マシンの停止時間の比較

仮想マシンの停止中に転送されたメモリページ数と仮想マシンの停止時間を提案手法と既存手法で比較し，5.2.1 項での QoS スコアの向上が仮想マシンの停止時間の差異によって得られたものではなく，移送タイミングが適切であったために得られたものであることを確認する．提案手法を用いて仮想マシンの移送を行う場合，コネクションの状態によって仮想マシンの移送の実行を延期することがあるため，既存手法を用いた場合よりもメモリページの再送回数が多くなり，仮想マシンを停止させた後の最後のメモリ転送の際に送るページ数が少なくなる可能性がある．仮想マシンの停止中の転送ページ数が少なくなると仮想マシンの停止時間が短縮される．すると，提案手法を用いた場合，適切な移送タイミングを見つけられているかどうかにかかわらず，既存手法よりもベンチマークのスコアが向上する可能性がある．そこで，提案手法を用いた場合と既存手法を用いた場合で仮想マシンの停止中に転送されたメモリページ数と仮想マシンの停止時間を比較し，5.2.1 項での QoS スコアの向上は仮想マシンの停止時間の差異によるものではないことを示す．

移送経路の帯域を 100 MBps に設定して実験した場合の，仮想マシン停止中の平均転送メモリページ数と仮想マシンの平均停止時間を**表 2** に表す．表より，5.2.1 項で表した提案手法の QoS スコアの高さは仮想マシンの停止時間が短かったためではなく，移送タイミングを適切に決定したためであることが示された．仮想マシンの停止中に転送したメモリページ数は提案手法を用いた場合は平均 48,197 ページ，既存手法を用いた場合は平均 46,762 ページとなり，提案手法を用いた用いた方が 1,400 ページ多くなった．この結果に比例するように，仮想マシンの停止時間は提案手法を用いた場合は 17.61 秒，既存手法を用いた場合は 17.20 秒となり，提案手法を用いた方が 0.41 秒長くなるという結果が得られた．

5.2.4　メモリの転送にかかる時間の比較

移送プロセス中のメモリの転送にかかる時間を提案手法と既存手法と比較する．提案手法を用いた場合，コネクションの状態が移送に適した状態になるタイミングでメモリの転送を中止し仮想マシンの移送を行うので，移送を行う前のメモリの転送回数には実験によりばらつきがある．5.2.1 項で行った 100 Mbps の帯域上での提案手法を用いた移送実験のうち，両手法が要した転送時間を**表 3** に，転送処理において転送したメモリ量を**図 7**，**図 8** に示す．

表 3　転送時間の比較
Table 3　Trasfer time.

既存手法（sec）	提案手法（sec）
157.6	330.9

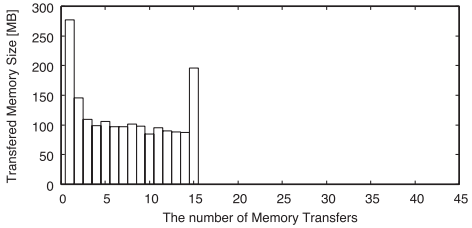


図 7　既存手法のメモリ転送回数と転送メモリ量
Fig. 7　Transferred memory size and memory transfer iterations (default VM migration).

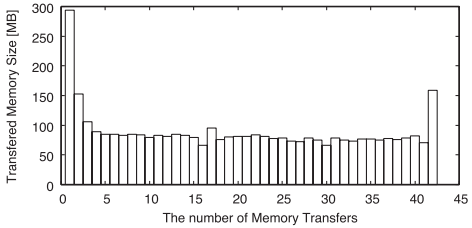


図 8　提案手法のメモリ転送回数と転送メモリ量
Fig. 8　Transferred memory size and memory transfer iterations (proposal VM migration).

表 3 より，実際にメモリ転送に要する時間は提案手法を用いた場合の約 2.1 倍となっていることが分かる．一方，図 7，図 8 より，提案手法の転送回数が既存手法にくらべて約 2.6 倍に増大していることが分かる．しかし，後半になると 1 度に転送されるメモリ量は少なくなるため，実際にメモリ転送に要する時間は前半より短くなる．よって，本評価における最

表 4　平均応答時間の比較
Table 4　Average response time.

	平均応答時間（sec）	
	4 MB 以上のファイルの転送	4 MB 未満のファイルの転送
提案手法	150.59	4.07
既存手法	149.58	5.24

悪な場合でも，転送時間は約 2 倍程度に抑えられることが分かった．

5.2.5　平均応答時間の比較

3 章で述べたとおり，提案手法は長時間のデータの転送を行っているコネクションを移送に適した状態のコネクションであると判断する．そこで，提案手法が実際に長時間のデータ転送を行っているコネクションを発見し移送タイミングとして認識できていることを確認する．先述の SPECweb を用いた実験の結果からリクエストごとの平均応答時間を入手し，比較する．

表 4 は，100 Mbps の帯域を使用して提案手法と既存手法における 4 MB 以上のファイルの転送の平均応答時間と 4 MB 未満のファイルの転送の平均応答時間を示す．4 MB 未満のファイルの転送を行うリクエストの場合，提案手法を用いることで約 1.17 秒既存手法よりも応答が早くなっている．また，提案手法では 4 MB 以上のファイルの転送の平均応答時間は既存手法を用いた場合よりも約 1.01 秒長くなっており，移送によってこれらのファイルの転送の中断が起こっていることが確認できる．表 4 より，4 MB 以上のファイルの転送にかかる時間は 4 MB 未満のファイルの転送にかかる時間よりもはるかに長いので，提案手法を用いたことによって 4 MB 以上のファイルの転送の応答時間が 1.01 秒増加してもクライアントに与える影響は相対的に少ないと考えられる．

SPECweb の Support ワークロードにおいて 4 MB 未満の通信の割合は全体の 96.6%を占め，一方で大量のデータの転送を行っているのは 3.4%である．よって，提案手法を用いることで大多数のコネクションの応答性能を 1.17 秒改善できているといえる．

5.2.6　提案手法による Web サーバの性能低下

提案手法を用いた場合に発生するオーバヘッドによって，仮想マシン上の Web サーバの性能が通常に比べてどの程度低下するかを評価する．提案手法のプロトタイプは Dom-0 上で仮想マシンの通信パケットからアプリケーション層プロトコルメッセージを復元して解釈する処理を行っている．そのため，通常よりも Dom-0 が行う処理が増え，その影響で他の仮想マシン上のサーバの性能が低下することが予測される．メッセージの処理によって起こるサーバ性能の低下を評価するために，Dom-0 上でメッセージの復元と解釈を行った場合とそれらを行わなかった場合で，SPECweb のスコアを比較した．メッセージ処理のオーバヘッドだけを評価するために本実験では仮想マシンの移送は行わなかった．実験に使用した SPECweb の設定はすべて 5.2.1 項の実験と同じである．

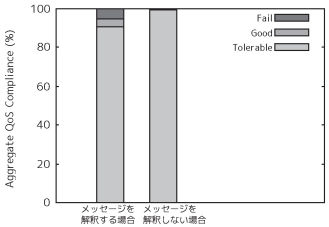


図 9　仮想マシン上のサーバの性能低下
Fig. 9　Overhead.

図 9 はメッセージの復元と解釈を行った場合とそれらを行わなかった場合での Web サーバの QoS スコアを表したものである．メッセージの復元と解釈を行わない場合，Tolerable に分類される応答が 99.6%だったのに対し，復元と解釈を行った場合は 91.0%であった．

ただし，メッセージの復元と解釈によるオーバヘッドは，不必要なときにメッセージを復元する機構を停止させておくことで減らすことができる．WAN 上での仮想マシンの移送プロセスには長い時間がかかるので，移送プロセスを開始する際にメッセージ解釈機構を起動させれば移送タイミングの判断時までにはほぼすべてのコネクションの状態を把握することができる．よって，メッセージの解釈機構を動かすのは移送プロセスが始まってからで十分であり，それ以外の状態ではメッセージ解釈機構を停止させておくことでオーバヘッドを抑えることができる．

6. ま と め

本論文では仮想マシンの移送によるダウンタイムをクライアントに感知されにくくする手法を提案した．本手法は，サーバとクライアント間でやりとりされるアプリケーション層プロトコルに着目してコネクションの状態を追跡する．各コネクションの状態がクライアントに仮想マシンのダウンタイムを感知されにくい状態になった時点で仮想マシンを移送する．

提案手法を用いて WAN 環境を再現したネットワーク上で仮想マシンの移送を行い，サーバが提供するサービスの品質を提案手法を用いた場合と比較した．提案手法を用いて移送を

行った場合，既存手法を用いて移送した場合よりも高い QoS スコアがベンチマークより得られた．

今後は，コネクションの状態を考慮しながら複数の仮想マシンを同時に移送する手法を研究する必要がある．本論文の実験では Web サーバとバックエンドシミュレータを同じ仮想マシン上で動作させて移送を行ったが，実際にはバックエンドサーバは別々の仮想マシン上で動作していることが多い．2 台以上で仮想マシンサーバの移送を行う場合，いかにサービスの停止時間が短くなるような手順で移送を行うかが重要となる．また，本論文で提案手法に考慮させたプロトコルは HTTP のみであるため，FTP などの他のプロトコルを用いた際に提案手法が有効であるかを評価する必要がある．

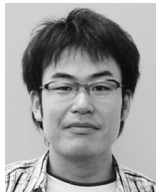
参 考 文 献

1) AFCOM. 2009/2010 AFCOM Data Center Trends Survey Results & Analysis (2010).
2) 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, *Proc. ACM Symposium on Operating Systems Principles*, pp.164–177 (2003).
3) Bradford, R., Kotsovions, E., Feldmann, A. and Schiöberg, H.: Live Wide-Area Migration of Virtual Machines Including Local Persistent State, *Proc. ACM International Conference on Virtual Execution Environments*, pp.169–179 (2007).
4) Clark, C., Fraser, K., Hand, S., Hansen, J.G., Jul, E., Limpach, C., Pratt, I. and Warfield, A.: Live Migration of Virtual Machines, *Proc. 2nd USENIX Symposium on Networked Systems Design and Implementation*, pp.273–286 (2005).
5) Goldberg, R.P.: Survey of Virtual Machine Research, *IEEE Computer Magazine*, Vol.7, pp.34–45 (1974).
6) Hines, M.R. and Gopalan, K.: Post-Copy Based Live Virtual Machine Migration Using Adaptive Pre-Paging and Dynamic Self-Ballooning, *Proc. ACM International Conference on Virtual Execution Environments*, pp.51–60 (2009).
7) Standard Performance Evaluation Corporation: The SPECweb2005 benchmark. http://www.spec.org/web2005/
8) Wood, T., Shenoy, P., Venkataramani, A. and Yousif, M.: Black-box and Gray-box Strategies for Virtual Machine Migration, *Proc. 4th USENIX Symposium on Networked Systems Design and Implementation*, pp.229–242 (2007).
9) 河野健二，品川高廣，ラハトカビル：TCP ストリームに対するフィルタリングによるインターネット・サーバの安全性向上，情報処理学会論文誌：コンピューティングシステム，Vol.46, No.SIG4(ACS9), pp.33–44 (2005).

(平成 22 年 7 月 26 日受付)
(平成 22 年 11 月 24 日採録)

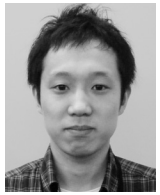
石川　豊

1985 年生まれ．2008 年慶應義塾大学理工学部情報工学科卒業．2010 年慶應義塾大学大学院理工学研究科開放環境科学専攻前期博士課程修了．仮想マシン技術，オペレーティングシステム等のシステムソフトウェアに興味を持つ．



山田　浩史（正会員）

1981 年生まれ．2004 年電気通信大学電気通信学部情報工学科卒業．2009 年慶應義塾大学大学院理工学研究科開放環境科学専攻後期博士課程修了．現在，慶應義塾大学理工学部特別研究助教．博士（工学）．2008 年，2009 年度情報処理学会論文賞，2008 年度山下記念研究賞受賞．仮想マシン技術，オペレーティングシステム等のシステムソフトウェアに興味を持つ．ACM，USENIX，IEEE/CS 各会員．



浅原　理人（正会員）

2005 年電気通信大学電気通信学部情報工学科卒業．2007 年慶應義塾大学大学院理工学研究科開放環境科学専攻修士課程修了．2010 年慶應義塾大学大学院理工学研究科開放環境科学専攻後期博士課程所定単位取得満期退学．現在，日本電気株式会社サービスプラットフォーム研究所勤務．クラウドコンピューティング，Peer-to-Peer システム，ミドルウェア，オペレーティングシステムに興味を持つ．IEEE/CS，ACM，USENIX 各会員．

花岡　美幸（正会員）

2005 年電気通信大学電気通信学部情報工学科卒業．2007 年慶應義塾大学大学院理工学研究科開放環境科学専攻修士課程修了．2010 年慶應義塾大学大学院理工学研究科開放環境科学専攻後期博士課程単位取得満期退学．同年より，（株）日立製作所中央研究勤務．システムソフトウェア，インターネットセキュリティに興味を持つ．IEEE/CS，ACM，USENIX 各会員．

河野　健二（正会員）

1993 年東京大学理学部情報科学科卒業．1997 年東京大学大学院理学系研究科情報科学専攻博士課程中退，同専攻助手に就任．現在，慶應義塾大学理工学部情報工学科准教授．博士（理学）．1999 年度，2008 度，2009 年度情報処理学会論文賞受賞．2000 年度山下記念研究賞受賞．オペレーティングシステム，システムソフトウェア，ディペンダブルシステム，インターネットセキュリティ等に興味を持つ．IEEE/CS，ACM，USENIX 各会員．