シャドウデバイスを用いた帯域外リモート管理を継続可能なVMマイグレーション

鵜木 智矢[1]　光来 健一[1]

概要：IaaS 型クラウドが提供する帯域外リモート管理にはクラウドの管理者への情報漏洩の危険性があるため，その対策として VSBypass と呼ばれるシステムが提案されている．VSBypass では，信頼できない管理者が管理権限を持つ仮想化システムの外側でシャドウデバイスを動作させ，VM に対する帯域外リモート管理の入出力をシャドウデバイスで安全に処理する．しかし，VM マイグレーションにより VM を異なるホストに移動させた場合，シャドウデバイスの状態が失われ，シャドウデバイスを用いた帯域外リモート管理が行えなくなる．そこで本研究では，シャドウデバイスの状態を移送先に転送できるようにすることにより，VM マイグレーション後もシャドウデバイスを用いた帯域外リモート管理を継続可能にするシステム USShadow を提案する．USShadow では，仮想化システム内部の移送クライアントが仮想化システムの外側にあるシャドウデバイスの状態を取得することを可能にする．この状態はシャドウデバイスにおいて暗号化・復号化されるため，マイグレーション中の情報漏洩を防ぐことができる．我々は，USShadow を Xen に実装し，マイグレーション時間やダウンタイムの増加が小さいことを確認した．

1. はじめに

クラウドコンピューティングの一形態である IaaS 型クラウドにおいて，ユーザはネットワーク経由で仮想マシン（VM）を利用でき，他のクラウドサービスと比較して自由度の高いシステムの構築が可能である．IaaS 型クラウドでは，ユーザは帯域外リモート管理と呼ばれる管理手法を用いて提供された VM の管理を行う．帯域外リモート管理では，VM の仮想デバイスに直接アクセスすることにより VM 内のシステムを管理することができるため，VM 内のネットワーク設定に依存しないという利点がある．

その一方で，帯域外リモート管理ではクラウドの管理者によって機密情報が盗まれる危険性がある．実際に，管理者の約 35 %が機密情報に無断でアクセスしたことがあり [1]，サイバー犯罪の約 28 %が内部犯行である [2] という報告もある．このようにクラウドの管理者の中には信頼できない管理者も存在する．クラウド管理者は VM の仮想デバイスにアクセスすることができるため，帯域外リモート管理におけるユーザの入出力を仮想デバイスを通して容易に盗聴することができる．

仮想デバイスからクラウド管理者への情報漏洩を防ぐために VSBypass [3] と呼ばれるシステムが提案されている．VSBypass はネストした仮想化と呼ばれる技術を用いることにより，従来の仮想化システムを VM の中で動作させる．そして，仮想化システムの外側でシャドウデバイスと呼ばれる仮想デバイスを用いて帯域外リモート管理を行うことにより，仮想化システムの管理権限を持つクラウド管理者への情報漏洩を防ぐ．しかし，VSBypass では VM マイグレーションの際にシャドウデバイスの状態が失われるため，マイグレーション後にシャドウデバイスを用いた帯域外リモート管理を継続することができない．さらに，帯域外リモート管理中に VM マイグレーションを行うと入出力情報が失われる可能性もある．

そこで本稿では，シャドウデバイスの状態を移送先に転送できるようにすることで，マイグレーション後もシャドウデバイスを用いた帯域外リモート管理を継続することができるシステム USShadow を提案する．USShadow では，マイグレーションの際に仮想化システム内の移送クライアントが仮想化システム外部のシャドウデバイスと通信を行い，シャドウデバイスの状態を取得する．そして，移送先ホストでは移送サーバがシャドウデバイスにその状態を復元する．この際に，シャドウデバイスの状態にはユーザの入出力が含まれる可能性があるため，クラウド管理者への情報漏洩を防ぐ必要がある．そこで，USShadow では移送元のシャドウデバイスにおいて状態の暗号化を行い，移送先のシャドウデバイスで状態の復号を行う．仮想化システ

[1] 九州工業大学
Kyushu Institute of Technology

ム内の信頼できない管理者はシャドウデバイスにアクセスすることはできないため，情報漏洩を防止しながら安全にシャドウデバイスの状態を転送することができる．

我々は USShadow を Xen 4.4 に実装した．シャドウデバイスと安全に通信を行うために，USShadow ではウルトラコールと呼ばれる機構を用いて，仮想化システムの下で動作するクラウドハイパーバイザを直接呼び出す．クラウドハイパーバイザ経由でシャドウデバイスを呼び出し，移送クライアントや移送サーバ（合わせて移送マネージャと呼ぶ）のバッファに直接アクセスすることで効率のよい通信を実現する．そのために，クラウドハイパーバイザは移送マネージャのページテーブルを用いて，移送マネージャのバッファの仮想アドレスをクラウドハイパーバイザが管理する物理アドレスに変換する．USShadow を用いてマイグレーション性能を測定する実験を行い，マイグレーションにかかる時間とダウンタイムの増加は従来のネストした仮想化システムおよび VSBypass に比べて小さいことが分かった．

以下，2 章でシャドウデバイスを用いた帯域外リモート管理と VM マイグレーションにおける問題について述べる．3 章で VM マイグレーション後にシャドウデバイスを用いた帯域外リモート管理を継続可能なシステム USShadow を提案する．4 章で USShadow の実装について述べ，5 章で USShadow のマイグレーション性能を調べた実験について述べる．6 章で関連研究に触れ，7 章で本稿をまとめる．

2. シャドウデバイスを用いた帯域外リモート管理

2.1 VSBypass

クラウドにおいて安全な帯域外リモート管理を実現するために，VSBypass と呼ばれるシステムが提案されている．VSBypass では，仮想化システムの外側でシャドウデバイスと呼ばれる仮想デバイスを動作させ，シャドウデバイス経由で帯域外リモート管理を行うことを可能にする．ユーザは仮想化システム内部の仮想デバイスを経由せずに仮想シリアルコンソールなどへの入出力を行えるため，仮想デバイスからの情報漏洩を防ぐことができる．VSBypass では，仮想化システム全体を信頼できないものとし，その外部だけを信頼できると仮定している．このようなシステムを実現するために，VSBypass ではネストした仮想化を用いて従来の仮想化システム全体を VM 内で動作させる．また，強制パススルーと呼ばれる機構を提供して，VM がシャドウデバイスを用いて入出力を行えるようにする．強制パススルー機構は VM が仮想デバイスに対して実行した入出力処理命令を横取りし，シャドウデバイスで透過的に処理する．

VSBypass のシステム構成を図 1 に示す．従来の仮想化システムはネストした仮想化を用いてクラウド VM と呼ばれる VM 内で動作する．クラウド VM の中ではユーザの VM（ユーザ VM）が複数動作する．また，従来の仮想デバイスもクラウド VM 内で動作する．クラウド VM の外側では安全な帯域外リモート管理に用いるシャドウデバイスがユーザ VM ごとに動作する．ユーザは SSH などを用いてネットワーク経由でシャドウデバイスにアクセスすることにより，帯域外リモート管理を行う．クラウド VM とシャドウデバイスの下ではハイパーバイザが動作し，このハイパーバイザをクラウドハイパーバイザと呼ぶ．

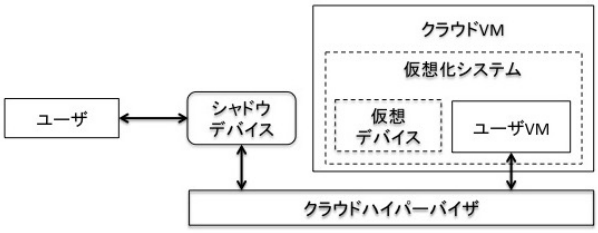

図 1 VSBypass のシステム構成

VSBypass では以下のようにしてユーザ VM の入出力の処理を行う．ユーザ VM が仮想デバイスに対して入出力命令を実行すると，通常は仮想化システム内で処理されるが，VSBypass ではその命令をクラウドハイパーバイザが横取りする．横取りした入出力命令は強制パススルー機構を用いてシャドウデバイスに転送して処理する．入力命令の場合には，リモートユーザが帯域外リモート管理によりシャドウデバイスに送信した入力をユーザ VM に返す．出力命令の場合は，ユーザ VM の出力をシャドウデバイスに書き込み，ユーザはそれを取得する．一方，シャドウデバイスで仮想割り込みが発生した時，シャドウデバイスはユーザ VM に仮想割り込みを転送する．その過程で信頼できない仮想化システムを経由するが，仮想割り込みには機密情報が含まれていないため情報漏洩の恐れはない．

2.2 VSBypass における VM マイグレーション

VSBypass における VM マイグレーションは，移送元ホストと移送先ホストの仮想化システム内でそれぞれ動作する移送クライアントと移送サーバの間で行われる．VM マイグレーションの流れは以下のようになる．まず，移送元ホストの移送クライアントがユーザ VM のメモリを移送先ホストの移送サーバに送信する．VM マイグレーションを行っている間にも VM の内容は更新されているため，更新されたメモリを再送する．再送するメモリ量が十分小さくなったところでユーザ VM を停止し，CPU や仮想デバイスの状態を移送サーバに送信する．移送先ホストでは，受信した CPU や仮想デバイスの状態を復元してユーザ VM の実行を再開する．

しかし，VSBypass では VM マイグレーションを行うと，マイグレーション後にシャドウデバイスを用いた帯域外リ

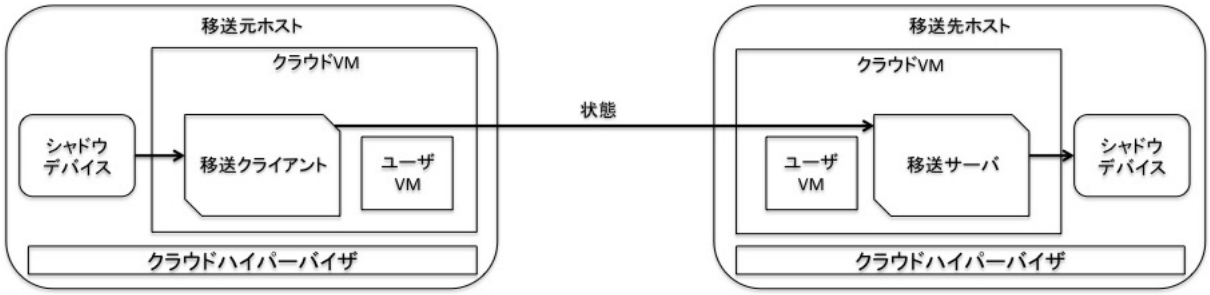

図 2 USShadow におけるマイグレーションの流れ

モート管理が行えなくなる．これは，シャドウデバイスが仮想化システムの外側で動作するため，従来の仮想デバイスのように状態を保存したり復元したりすることができないためである．そのため，シャドウデバイスの状態を移送先ホストに転送することができず，シャドウデバイスの状態が失われてしまう．その結果，ユーザ VM やリモートユーザからシャドウデバイスに正常にアクセスできなくなる可能性がある．

3. USShadow

本稿では，VM マイグレーション後もシャドウデバイスを用いた帯域外リモート管理を継続可能にするシステム USShadow を提案する．USShadow では，VM マイグレーションの際に仮想化システム内の移送マネージャが仮想化システムの外側にあるシャドウデバイスの状態を転送することを可能にする．USShadow における VM マイグレーションの流れを図 2 に示す．USShadow では，移送元ホストの移送クライアントがシャドウデバイスと通信を行い，シャドウデバイスの状態を取得する．そして，取得したシャドウデバイスの状態を移送先ホストの移送サーバに送信し，移送元ホストのシャドウデバイスを停止させる．一方，移送先ホストの移送サーバでは VM マイグレーションを開始したらシャドウデバイスを起動する．そして，移送クライアントからシャドウデバイスの状態を受信した後，起動したシャドウデバイスと通信を行い受信した状態をシャドウデバイスへ送る．このように，シャドウデバイスの状態を保存・復元することで，マイグレーション後に正常にシャドウデバイスを用いることが可能になる．

従来，マイグレーションの際に行われる仮想デバイスの状態の保存・復元にはプロセス間通信が用いられていた．しかし，プロセス間通信を行うには移送マネージャと仮想デバイスが同一の OS 上で動作している必要がある．シャドウデバイスは仮想化システムの外側で動作しているため，移送マネージャとプロセス間通信を行うことはできない．そこで，ネットワークを用いた通信が考えられるが，ネットワーク通信を行うためにはシャドウデバイスに対してネットワーク経由でのアクセスを許可する必要がある．そのため，信頼できない仮想化システムからの攻撃を受ける危険性が増大する．また，ネットワーク通信を行うにはシステム内の多くのコンポーネントを経由する必要があるため，オーバヘッドが大きい．

そこで，USShadow では図 3 のように移送マネージャがクラウドハイパーバイザを直接呼び出すことによりシャドウデバイスとの通信を行う．そのために，USShadow はウルトラコールと呼ばれる機構を提供する．ウルトラコールは，CPU の機能を利用することで仮想化システムをバイパスしてクラウドハイパーバイザに直接制御を移す機構である．そのため，仮想化システムへの改変が不要であり，既存の仮想化システムを用いることができる．その後，クラウドハイパーバイザがシャドウデバイスを呼び出すと，シャドウデバイスは移送マネージャのメモリに直接アクセスすることによりデータをやりとりする．そのために，クラウドハイパーバイザは移送クライアントのバッファの仮想アドレスをクラウド VM の物理アドレスに変換する．シャドウデバイスはクラウドハイパーバイザの機能を用いてそのメモリを共有してアクセスする．

USShadow では，シャドウデバイスの状態はシャドウデバイス自身が暗号化・復号化を行う．これは，移送マネージャにおいてシャドウデバイスの状態からの情報漏洩を防ぐためである．シャドウデバイスの状態にはユーザの入力やユーザ VM からの出力が含まれる可能性がある．シャドウデバイスは信頼できない仮想化システムの外側で動作するため，シャドウデバイスで暗号化・復号化を行えば情報が漏洩することはない．移送クライアントは暗号化されたシャドウデバイスの状態を取得し，それを移送サーバに送信する．移送サーバは暗号化された状態をシャドウデバイスに送信し，シャドウデバイスにおいてその状態が復号さ

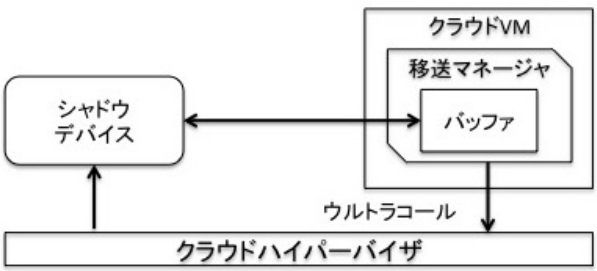

図 3 シャドウデバイスとの通信

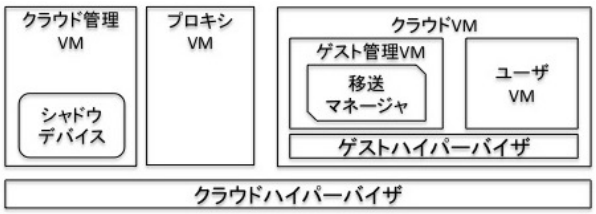

図 4　USShadow のシステム構成

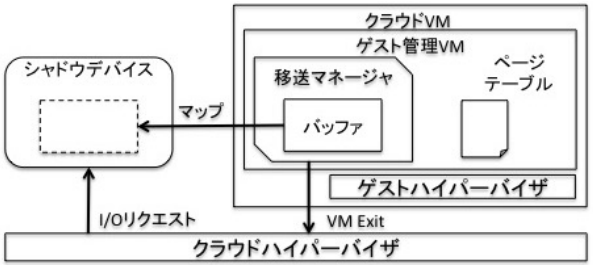

図 5　シャドウデバイスとの通信の流れ

れる．

4. 実装

我々は USShadow を Xen 4.4 をベースとする VSBypass に実装した．

4.1 システム構成

USShadow のシステム構成は VSBypass と同様である．図 4 のように，Xen を用いた仮想化システム内では，ユーザ VM に加えて管理 VM が動作しており，仮想デバイスと移送マネージャは管理 VM 内で動作する．管理 VM とユーザ VM の下ではハイパーバイザが動作する．これらの管理 VM およびハイパーバイザをそれぞれゲスト管理 VM，ゲストハイパーバイザと呼ぶ．また，クラウドハイパーバイザの上ではクラウド VM に加えてクラウド管理 VM が動作し，シャドウデバイスはクラウド管理 VM 内で QEMU の一部として動作する．シャドウデバイスを用いた帯域外リモート管理を容易にするために，ユーザ VM ごとにプロキシ VM と呼ばれる VM を動作させ，その仮想デバイスをシャドウデバイスとして用いる．プロキシ VM には必要最低限のメモリやディスクおよび，ユーザ VM と同数の仮想 CPU を割り当てる．

4.2 シャドウデバイスを用いた帯域外リモート管理

USShadow におけるシャドウデバイスを用いた帯域外リモート管理は VSBypass と同様に行われる．ユーザはプロキシ VM の仮想シリアルデバイスをシャドウデバイスとしてアクセスすることにより，透過的に仮想化システム内の対応するユーザ VM にアクセスすることができる．クラウドハイパーバイザにおいてプロキシ VM とユーザ VM の二つの VM を対応づけるために，ユーザ VM の拡張ページテーブル（EPT）を利用する．EPT はユーザ VM ごとに作成され，EPT ポインタがユーザ VM の仮想 CPU 内の VMCS に格納されている．クラウドハイパーバイザは EPT ポインタに基づいてユーザ VM を識別し，プロキシ VM に結びつける．また，管理するユーザ VM をユーザが識別できるようにするために，ユーザ VM 内でウルトラコールを実行して VM タグと呼ばれる識別子を登録しておく．

USShadow では，ユーザ VM が仮想シリアルデバイスに対して入出力命令を実行した時だけその命令を横取りする．入出力命令を実行すると VM Exit が発生し，クラウドハイパーバイザに制御が移る．仮想シリアルデバイスのポート番号へのアクセスだった場合，クラウドハイパーバイザは，シャドウデバイスに入出力を転送する．転送先のシャドウデバイスはユーザ VM の EPT から対応するプロキシ VM を見つけて決定する．シャドウデバイスで入出力処理が完了すると，そのシャドウデバイスを提供しているプロキシ VM に対応するユーザ VM を見つけて，VM Entry を実行する．シャドウデバイスで発生した仮想割り込みはゲスト管理 VM 内で動作する割り込みサーバに転送し，ゲストハイパーバイザ経由でユーザ VM に挿入する．

4.3 シャドウデバイスとの通信

ゲスト管理 VM 内の移送マネージャがシャドウデバイスと通信する際にも，ユーザ VM からの入出力命令の横取りと同様に，クラウドハイパーバイザを経由する．入出力命令と異なり，移送マネージャはウルトラコールを用いて図 5 のように VM Exit を発生させることでクラウドハイパーバイザを明示的に呼び出す．ウルトラコールはハイパーコール実行にも用いられている vmcall 命令を用いて実装した．クラウド VM 内で vmcall 命令を実行すると，ゲストハイパーバイザではなく，まずクラウドハイパーバイザにトラップされるという仕組みを利用している．このウルトラコールでは引数として，状態の保存・復元のどちらを行うか，シャドウデバイスの状態が格納される移送マネージャのバッファの仮想アドレス，そのサイズをクラウドハイパーバイザに渡す．クラウドハイパーバイザに渡すバッファは，ゲスト管理 VM において mmap システムコールを用いて確保した 1 ページ分の匿名メモリであり，スワップアウトされないように mlock システムコールによってロックする．

ウルトラコールで呼び出されたクラウドハイパーバイザは，シャドウデバイスがゲスト管理 VM のメモリにアクセスできるようにするために，バッファ用メモリのアドレス変換を行う．まず，クラウド VM の仮想 CPU からページテーブルのアドレスが格納されている CR3 レジスタの値を取得する．ゲスト管理 VM では CPU が準仮想化されているため，VM Exit が発生した時のクラウド VM の仮

想 CPU のレジスタ値はゲスト管理 VM の仮想 CPU のものと同一である．そして，移送マネージャのページテーブルをたどり，移送マネージャの仮想アドレスからクラウド VM の物理アドレスへの変換を行う．ゲスト管理 VM ではメモリも準仮想化されているため，ページテーブルにはクラウド VM の物理アドレスが直接格納されており，EPT をたどる必要はない．

クラウドハイパーバイザはシャドウデバイスに特殊な I/O リクエストを送信することにより，シャドウデバイスを呼び出す．通常の I/O リクエストと区別するために，専用の I/O ポート番号を利用する．この際に，I/O リクエストを送信するシャドウデバイスを特定できるようにする必要があるが，現在の実装ではあらかじめ決められたシャドウデバイスを呼び出している．今後，状態を保存・復元するシャドウデバイスを指定できるようにする必要がある．シャドウデバイスでリクエスト処理が完了したら，処理結果をウルトラコールの返り値として移送マネージャに返す．

I/O リクエストを受信したシャドウデバイスは，ゲスト管理 VM のメモリをマップすることにより移送マネージャと直接通信を行う．シャドウデバイスはプロキシ VM 用の QEMU 内に実装されているが，この QEMU はプロキシ VM のメモリにアクセスするインタフェースしか提供していない．そのため，クラウドハイパーバイザの機能を用いてクラウド VM のメモリにアクセスするインタフェースを追加した．

4.4 シャドウデバイスの状態の保存・復元

仮想デバイスの状態を保存・復元するために QEMU が提供している機能を用いて，シャドウデバイスの状態の保存・復元を実装した．従来の QEMU は仮想デバイスの状態をファイルかソケットにしか保存することができなかった．この機能は，VM の状態をディスクに保存したり，マイグレーションで転送したりする際に使われるためである．USShadow では，仮想デバイスの状態をメモリに保存する機能を追加し，クラウド VM のメモリに保存できるようにした．これにより，QEMU にメモリマップされた移送マネージャのバッファにシャドウデバイスの状態を書き込んだり，読み込んだりすることができる．

USShadow では，様々な仮想デバイスの中で仮想シリアルデバイスの状態のみの保存・復元を行う．状態の保存は，デバイスレジスタの値などをバッファに保存することにより行う．その際に，デバイスレジスタの値と暗号鍵との排他的論理和をとることにより暗号化を行っている．AES などの強力な暗号化を用いたり，暗号鍵の管理を行ったりすることは今後の課題である．一方，状態の復元はバッファのデータを暗号鍵を用いて復号し，デバイスレジスタに書き込むことにより行う．

5. 実験

USShadow を用いて VM マイグレーション後の帯域外リモート管理の動作を確認し，VM マイグレーションの性能を調べる実験を行った．実験には，Intel Xeon E3-1226v3 の CPU，8GB のメモリ，ギガビットイーサネットを搭載したマシン 2 台を移送元ホスト，移送先ホストとして用いた．仮想化システムには Xen 4.4 を用い，クラウド VM には 1 個の仮想 CPU と 3GB のメモリ，ユーザ VM には 1 個の仮想 CPU と 256MB のメモリ，プロキシ VM にも 1 個の仮想 CPU と 512MB のメモリをそれぞれ割り当てた．比較として，ネストした仮想化を用いるがシャドウデバイスは用いない従来システムおよび，マイグレーション時にシャドウデバイスの状態を転送できない VSBypass を用いた．

5.1 動作確認

USShadow を用いて，VM マイグレーション後にシャドウデバイスを用いた帯域外リモート管理が行えるかの確認を行った．帯域外リモート管理として仮想シリアルコンソールを用い，ユーザ VM にログインした状態で VM マイグレーションを行った．VM マイグレーション後に，マイグレーション先のシャドウデバイスに接続して帯域外リモート管理を行ったところ，正常にアクセスでき，ユーザ VM にログインした状態から帯域外リモート管理を再開できた．

5.2 状態の保存・復元時間

USShadow において，シャドウデバイスの状態の保存・復元にかかる時間の測定を行った．それぞれの時間としてウルトラコールの実行にかかる時間を測定した．測定結果を図 6 に示す．シャドウデバイスの状態の保存に 4.4 ミリ秒，復元に 4.3 ミリ秒かかった．5.3 節に示すダウンタイム

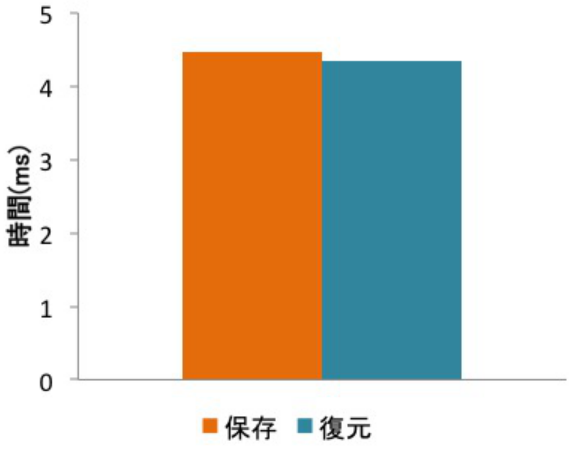

図 6 シャドウデバイスの保存・復元にかかる時間

に対して，状態の保存・復元にかかる時間は十分短いことが分かった．

5.3 マイグレーション性能

USShadow における VM マイグレーションにかかる時間を測定し，ネストした仮想化を用いた従来システムと VSBypass との比較を行った．実験結果を図 7 に示す．USShadow と VSBypass のマイグレーション時間の差は 0.2 秒であった．これはシャドウデバイスの状態を扱うためのオーバヘッドと考えられる．一方，従来のネストした仮想化システムと比べると，3 秒程度のマイグレーション時間の増加が見られた．これは，シャドウデバイスを用いた入出力処理を行うことによるオーバヘッドと考えられる．

次に，VM マイグレーション中のダウンタイムを図 8 に示す．ダウンタイムは VM が停止されてから再開されるまでの時間を測定した．USShadow は，VSBypass と比べてダウンタイムが 0.03 秒だけ増加し，従来のネストした仮想化システムに比べても 0.3 秒の増加であった．これらもシャドウデバイスの状態を扱うオーバヘッドおよびシャドウデバイスを用いた入出力処理のオーバヘッドによるものと考えられる．

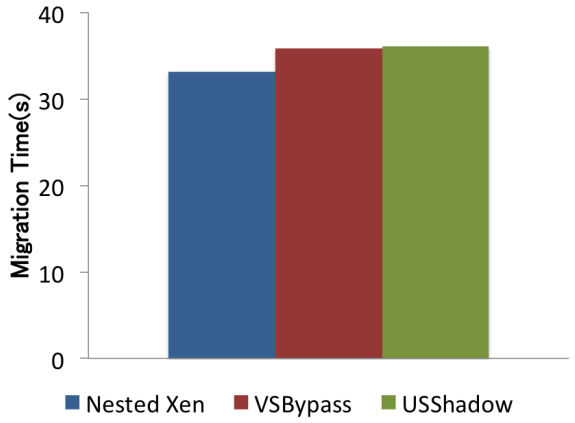

図 7 マイグレーション時間

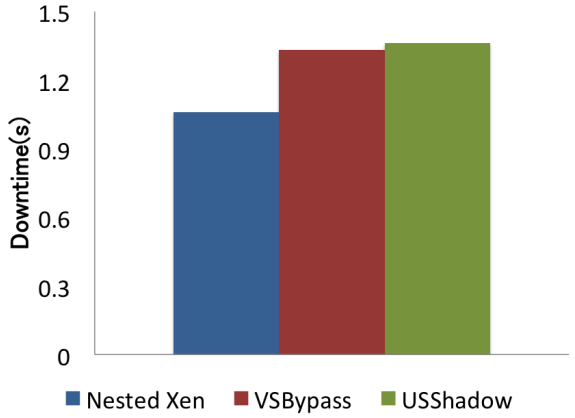

図 8 ダウンタイム

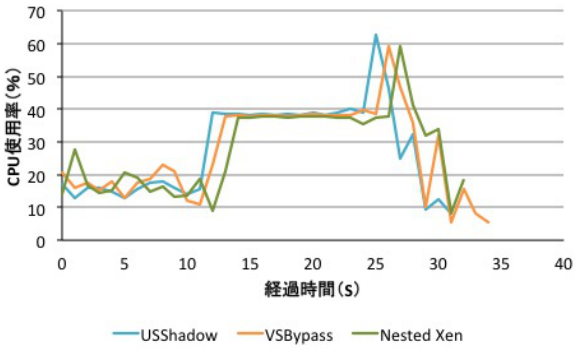

図 9 クラウド VM の CPU 使用率の変化

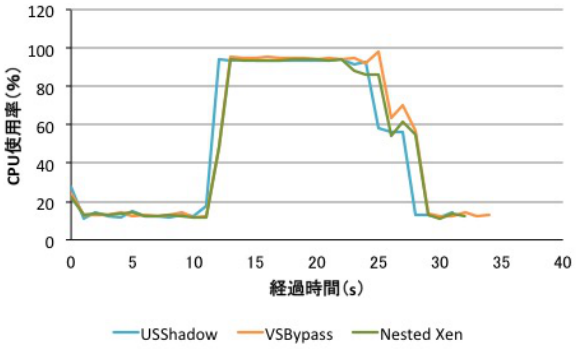

図 10 クラウド管理 VM の CPU 使用率の変化

5.4 CPU 使用率

USShadow でのマイグレーション中の CPU 使用率を測定し，ネストした仮想化を用いた従来システムと VSBypass との比較を行った．移送クライアントが動作するクラウド VM のマイグレーション中の CPU 使用率は図 9 のようになった．どのシステムでも CPU 負荷は同じように変化したが，USShadow ではマイグレーションの最終段階の CPU 使用率が他の二つのシステムより約 3 %高くなった．これはシャドウデバイスの状態を転送するオーバヘッドと考えられる．図 10 はクラウド管理 VM における CPU 使用率の変化である．どのシステムでも CPU 負荷は同程度であり，USShadow においてシャドウデバイスの状態を扱うオーバヘッドは特には見られなかった．

一方，図 11 はクラウド VM 内のゲスト管理 VM での CPU 使用率である．クラウド VM やクラウド管理 VM と同様の変化をしていることが分かる．ゲスト管理 VM の CPU 使用率がクラウド VM のものを超えているのは，クラウド管理 VM でネットワーク処理を行っている間も CPU を使用したとみなされているためと考えられる．

6. 関連研究

USShadow のシャドウデバイスは一種のパススルーデバイスである．パススルーデバイスを用いた VM をマイグレーションするシステムはいくつか提案されている．

ボンディングドライバを利用したマイグレーション [4] で

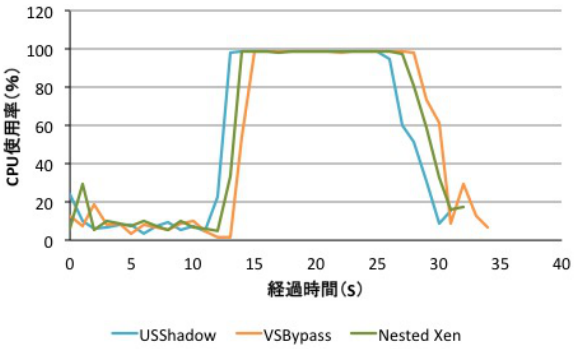

図 11 ゲスト管理 VM の CPU 使用率の変化

は，ボンディングドライバを用いてパススルーネットワークデバイスと準仮想化ネットワークデバイスを束ねる．マイグレーション時にパススルーデバイスをホットアンプラグすると，準仮想化デバイスに自動的に切り替わる．準仮想化デバイスを用いるデバイスは容易にマイグレーションすることができる．マイグレーション後にパススルーデバイスをホットプラグすると，準仮想化デバイスから自動的に切り替わる．設定の変更のみでマイグレーションに対応できる利点があるが，ネットワークデバイスにしか対応しておらず，ホットアンプラグの際にダウンタイムが発生するなどの欠点もある．

Network Plug-In Architecture (NPIA/NPA) では，マイグレーション中にネットワークドライバのプラグインを抜き差しすることでパススルーデバイスを用いる VM のマイグレーションに対応する．ボンディングドライバを用いる手法と同様に，マイグレーションの際には準仮想化デバイスのプラグインに切り替える．ボンディングドライバより切り替え時間が短いという利点があるが，SR-IOV NIC にしか対応しておらず，ネットワークドライバの大幅な書き換えが必要となる．

シャドウドライバを利用したマイグレーション [5] では，シャドウドライバが OS からネットワークドライバへのアクセスを記録する．パススルーデバイスを用いる VM のマイグレーション中はシャドウドライバがリクエストを処理し，移送先ホストで新たなネットワークドライバが起動したらリダイレクトを行う．しかし，ハイパーバイザと OS に大幅な変更が必要である．

CompSC [6] では，移送元ホストでパススルー NIC に行った操作を記録し，移送先ホストで再現することでデバイスレジスタの値を復元する．これにより，読むと値が消えるデバイスレジスタや，書き込めないレジスタ，書き込むとデバイスが何らかの処理を行うレジスタなどについても状態を復元することができる．加えて，CompSC ではハイパーバイザでエミュレーションを行い，統計情報等の復元も行う．USShadow ではパススルーデバイスであるシャドウデバイスは仮想デバイスであるため，状態を容易に保存・復元することができる．

SRVM [7] では，移送先ホストで SR-IOV NIC の状態を完全に復元するのではなく，正しく動作するようにだけ復元を行う．そのために，NIC が受信したパケットを書き込んだメモリを検出して，移送先ホストに転送できるようにしている．このようにして，SRVM では最小限の状態だけを復元する．

D-MORE [8] は VM マイグレーション後も従来の帯域外リモート管理を継続することができる．D-MORE では VM の仮想デバイスをマイグレーション可能でリモート管理専用 VM の上で動かす．マイグレーションの際には，ユーザ VM とリモート管理専用 VM を一緒に転送することで仮想デバイスの状態を保持する．しかし，シャドウデバイスを用いた帯域外リモート管理に D-MORE の手法を適用するには，シャドウデバイス専用の VM を仮想化システムの外部に用意する必要がある．仮想化システム内の移送クライアントがユーザ VM と一緒に仮想化システムの外側にある VM をマイグレーションできるようにすると，セキュリティ上の問題がある．

ネストした仮想化における通信の高速化も提案されている．Xen-Blanket [9] は，ゲスト管理 VM とクラウド管理 VM の間の高速な通信機構を提供する．Xen-Blanket では，ゲスト管理 VM の OS 内で Blanket ドライバを動作させ，ゲストハイパーバイザへのハイパーコール呼び出しを経由して，クラウド管理 VM と通信を行う．一方，USShadow ではゲスト管理 VM からクラウドハイパーバイザを直接呼び出して，クラウド管理 VM と通信を行う．Xen-Blanket ではクラウド VM 内の仮想化システムにのみ改変を加える設計であるのに対し，USShadow では仮想化システムは変更しない設計である点が異なる．

7. まとめ

本稿では，VM マイグレーション後にシャドウデバイスを用いた帯域外リモート管理を継続して行えるシステム USShadow を提案した．USShadow では，仮想化システムの外側にあるシャドウデバイスと安全で効率のよい通信を行い，シャドウデバイスの状態を移送先に転送できるようにする．マイグレーション中のシャドウデバイスからの情報漏洩を防ぐために，移送元のシャドウデバイスで状態を暗号化し，移送先のシャドウデバイスでその状態を復号する．USShadow を Xen 4.4 に実装し，シャドウデバイスを用いた帯域外リモート管理がマイグレーション後にも継続して行えることを確認した．また，USShadow による VM マイグレーションの際の性能低下は小さいことが分かった．

今後の課題は，KVM などの様々な仮想化システムに USShadow を対応させることである．すでに割り込み処理に関しては KVM に対応済みである．現在のところ，USShadow は仮想シリアルデバイスのサポートしか行って

いないため，仮想キーボードや仮想ビデオカードなどを用いた帯域外リモート管理への対応も行う予定である．また，複数のユーザ VM への対応や、より強力な暗号の利用も今後の課題である．

参考文献

[1] CyberArk Software. Global IT Security Service, 2009.
[2] PwC: US Cybercrime: Rising Risks, Reduced Readiness 2014.
[3] 二神翔太, 光来健一. VSBypass: ネストした仮想化を用いた VM の安全な帯域外リモート管理. SWoPP2016, 2016.
[4] E. Zhai, G. D. Cummings, and Y. Dong. Live migration with passthrough device for Linux VM. In Ottawa Linux Symposium, 2008.
[5] A. Kadav and M. M. Swift. Live migration of direct-access devices. SIGOPS Oper. Syst. Rev., 43:9-104, 2009.
[6] Z. Pan, Y. Dong, Y. Chen, L. Zhang, Z. Zhang: CompSC: live migration with pass-through devices. In Proc. VEE 2012, pp.109-120, 2012.
[7] X. Xu, B. Davda: SRVM: Hypervisor Support for Live Migration with Passthrough SR-IOV Network Devices. In Proc. VEE 2016, pp.65-77, 2016.
[8] Sho Kawahara,Kenichi Kourai: D-MORE: The Continuity of Out-of-band Remote Management across Virtual Machine Migration in Clouds, In Proc. Int. Conf. Utility and Cloud Computing, pp.176-185, 2014.
[9] D. Williams, H. Jamjoom, and H. Weatherspoon: The Xen-Blanket: Virtualize Once, Run Everywhere. In Proc. European Conference on Computer Systems, pp.113-126, 2012.