ネステッド仮想化の動的 ON/OFF による仮想マシンモニタ若化

安岡亮輔[†]　深井貴明[†]
品川高廣[††]　加藤和彦[†]

1. はじめに

仮想マシンモニタ（VMM）はサーバ用途で多く用いられているが，長時間連続的に稼働するとシステムの劣化，つまりソフトウェアエージング[1)] を引き起こすことがある．ソフトウェアエージングはパフォーマンスの低下や予期せぬシステムダウンを引き起こす要因となるため，これを予防するためには定期的にソフトウェア若化をおこなうことが有効である[1)]．ソフトウェア若化の典型例としては再起動があるが，サーバ用途で用いられている VMM の場合，再起動によるシステムダウンが与えるユーザへの影響は大きい．

仮想化環境で再起動によるシステムダウンを避ける手法としては，ライブマイグレーションがある．ライブマイグレーションでは，稼働中の仮想マシン（VM）を止めることなく別の物理マシンに移動することが出来る．そこで，再起動したい VMM 上で動作している VM を全て別のマシン上に移動することにより，サービスを停止せずに VMM を再起動できる．しかしマイグレーションは，VM の状態をネットワークを介して送るため，ネットワークに高い負荷がかかってしまう．

従来の研究では，ネステッド仮想化[2)] を用いて一台の物理マシン上で VMM を二つ動作させ，その間でライブマイグレーションをおこなうことでネットワーク転送のコストを削減する手法が提案されている[3)]．この手法はマイグレーションコストの削減には有効であるが，ソフトウェア若化をおこなわない時でも常にネステッド仮想化をおこなうことになるため，通常運用時のオーバーヘッドが大きくなってしまう．

本研究では，ネステッド仮想化を実行時に動的に ON/OFF できる軽量ハイパーバイザを用いることで，VMM の若化をおこなうときだけネステッド仮想化を ON にして VMM を二つ動作させ，VM のライブマイグレーション及び VMM の再起動による VMM 若化が終了したらネステッド仮想化を OFF にする手法を提案する．これにより，物理マシン間の VM マイグレーションによるコストは抑えつつ，通常運用時のネステッド仮想化のオーバーヘッドを削減することができる．また，VMM を二台動作させるためにはハードウェア分割を用いることで，VMM 若化を実現するためのハイパーバイザの構造を簡略化し，VMM 若化中のオーバーヘッドも削減する．

† 筑波大学
†† 東京大学

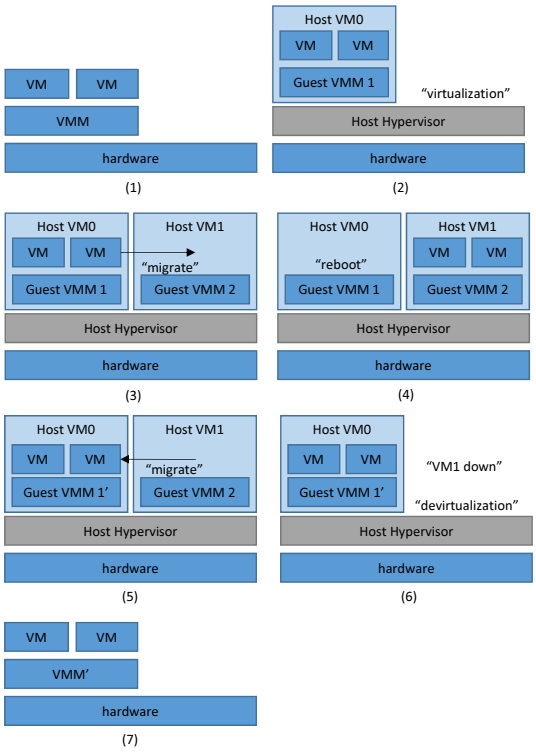

図 1　提案手法の概要

2. 概　　要

図 1 に提案手法の概要を示す．初期状態では，図 1(1) のように，Xen や KVM など従来の VMM といくつかの VM が hardware 上で動作している．VMM 若化をおこなう際には，まず図 1(2) のように，ホス

トとなるハイパーバイザを稼働中の VMM の下で動作させてネステッド仮想化を ON にする．このとき，従来の VMM は Guest VMM 1 となる．

次に，VM を退避させるための Guest VMM 2 を起動して，Guest VMM 1 と並行して動作させる．それから，図 1(3) のように Guest VMM 1 から Guest VMM 2 に VM をライブマイグレーションして，Guest VMM 1 の上で動作する VM がなくなるようにする．

VM の退避が完了すると，図 1(4) のように Guest VMM 1 を再起動して，VMM 若化をおこなうことが出来る．Guest VMM 1 の再起動が終わって新しい VMM（Guest VMM 1’）が動作し始めたら，図 1(5) のように，Guest VMM 2 上で動作していた VM を再び Guest VMM 1’ 上にライブマイグレーションして，元通り動作するようにする．

図 1(3) 及び (5) のライブマイグレーションは，一台の物理マシン上でおこなわれるため，ネットワークのコストを避けることが出来る．また，このとき VMBeam[3] のような最適化をおこなっても良い．ライブマイグレーションが終わったら，図 1(6) のように退避用の Guest VMM 2 を終了させる．その後，ホストとなるハイパーバイザでネステッド仮想化を OFF にして図 1(7) のようにハイパーバイザを終了させる．

図 1(1) 及び (7) に示すように，通常稼働時はネステッド仮想化が OFF になっているため，ネステッド仮想化のオーバーヘッドを削減することが出来る．

3. 実　　装

実装は，BitVisor[6] を改造して VM を二つ動作させられるようにした TinyVisor[7] をベースとする．TinyVisor は，CPU やメモリ，PCI デバイスを予め分割して各 VM に静的に割り当てることが出来る．CPU は Local APIC ID を用いて論理プロセッサ単位で特定し割り当て，メモリは 4KB 単位で物理アドレスで割り当てる範囲を指定し，PCI デバイスは Root Port 単位で割り当てる．このため，リソースの仮想化によるオーバーヘッドを削減して二つの VM が軽量に並行動作することが出来る．

本実装では，TinyVisor にネステッド仮想化を実装して，二つの VMM が並行して動作できるようにする．また，ネステッド仮想化を動的に ON/OFF する機能も実装する．

4. 関 連 研 究

VMBeam[3] では Xen のネステッド仮想化とライブマイグレーションを用いて，一台の物理マシン上で VMM を二つ動作させて VMM の若化をおこなう．この時二つの VMM が物理メモリを共有できることを利用して，ゼロコピーで VM のライブマイグレーションをおこなって高速化する．しかし，VMBeam ではホスト VMM として Xen を用いているため，通常稼働時にもネステッド仮想化のオーバーヘッドのほか，Xen そのもののオーバーヘッドもかかってしまう．

MicroVisor[4] では，OS のメンテナンス時に発生するダウンタイムを抑えるため，メンテナンス時のみ OS を仮想化して二つ動作させ，プロセスマイグレーションでアプリケーションを移動する．メンテナンス終了時には脱仮想化して通常稼働時のオーバーヘッドを削減する．提案手法では，OS ではなく VMM のソフトウェア若化をおこなう点が異なる．BMcast[5] も脱仮想化をおこなうことができる．提案手法は，ネステッド仮想化の ON/OFF をおこなう点が異なる．

5. お わ り に

本研究では，ネステッド仮想化の ON/OFF が可能な軽量ハイパーバイザを用いて，稼働中の VMM を若化することを可能にしつつ，通常稼働時における仮想化のオーバーヘッドを抑えられるシステムを提案した．現在 TinyVisor をベースとして VMM を二つ動かす実験をおこなっており，今後は ON/OFF 可能なネステッド仮想化の機能を実装していく予定である．

参　考　文　献

1) Y. Huang, et al. Software Rejuvenation: Analysis, Module and Applications. In Proc. 25th International Symposium on Fault-Tolerant Computing, pp. 381–390, 1995.
2) Muli Ben-Yehuda, et al. The Turtles Project: Design and Implementation of Nested Virtualization. In Proc. 9th OSDI, pp. 423–436, 2010.
3) K. Kourai, et al. Zero-copy Migration for Lightweight Software Rejuvenation of Virtualized Systems. In Proc. 6th APSys, 2015.
4) D. E. Lowell, et al. Devirtualizable Virtual Machines Enabling General, Single-Node, Online Maintenance. In Proc. 11th ASPLOS, pp. 211–223, 2004.
5) Y. Omote, et al. Improving Agility and Elasticity in Bare-metal Clouds. In Proc. 20th ASPLOS, pp. 145-159, Mar 2015.
6) T. Shinagawa, et al. BitVisor: A Thin Hypervisor for Enforcing I/O Device Security. In Proc. VEE 2009, pp. 121–130, Mar 2009.
7) Yuichi Watanabe. TinyVisor Project. `https://osdn.net/projects/tinyvisor/`.