15-IS03

次世代スーパーコンピューター向けの軽量な仮想計算機環境の実現に向けた研究開発

品川高廣（東京大学）

概要　我々が研究開発している軽量な仮想マシンモニタ「BitVisor」を応用して、次世代スーパーコンピューターのため軽量な仮想計算環境の実現に向けた研究開発をおこなう。近年のスパコンは多数ノードから構成されていることを活用し、ユーザに対してノード単位での割り当てをおこなうことにより、それぞれのノードのハードウェアはユーザが直接制御できるようにする。一方、各ノードのハードウェアへのアクセスは、仮想マシンモニタによって必要最小限の監視によりアクセス制御をおこなうことで、ノードのセキュリティ確保と多様な用途に応じたノード性能の最適化を両立することを目指す。今年度は、まずスパコンの各ノードに OS を高速にデプロイするシステムを TCP/IP に対応させた。また、セキュリティやライブマイグレーションなどの実現に関する研究開発をおこなった。

1.　共同研究に関する情報

(1) 共同研究を実施した拠点名

東京大学

(2) 共同研究分野

- □　超大規模数値計算系応用分野
- □　超大規模データ処理系応用分野
- □　超大容量ネットワーク技術分野
- ☑　超大規模情報システム関連研究分野

(3) 参加研究者の役割分担

副代表者の高野了成（産業技術総合研究所・情報技術研究部門）は、本システムにおいて高性能計算を実現するにあたって必要な助言などをおこなう。共同研究者の表祐志（筑波大学・大学院システム情報工学研究科）は、仮想マシンモニタを用いた実験に関する補助をおこなう。

2.　研究の目的と意義

我々が研究開発してきた軽量仮想マシンモニタ「BitVisor」を応用して、次世代スーパーコンピューターのための軽量な仮想計算環境の実現に向けた研究開発をおこなう。

近年のスパコンは、多数のノードから構成される集合体として実現されており、これらのノードの性能を最大限に引き出すことが重要となっている。一方で、スパコンのユーザやその用途はますます多様化してきており、単一のシステムで全ての要望を満たすことは難しくなりつつある。

例えば、現在のスパコンでは OS として Linux のみならず各種 UNIX や Windows も使われており、ユーザや用途によって最適な OS を使い分けたいなどの要望が増加することが考えられる。また、次世代スパコンではノード数が更に増加すると予想され、既存の汎用 OS ではスケールさせることが難しくなりつつある。また、用途別に特化した専用の軽量 OS を使いたいといった要望も出てくることが予想される。

しかし、ユーザが OS を自由に選択できるようにすると、ユーザ間でのセキュリティが問題となる。一般に OS は全てのハードウェア資源にアクセスすることが可能であり、他のユーザが利用するノードとの隔離や共有ストレージのセキュリティなどを確実に実施することが難しくなる。また、将来的に一般企業を含む様々なユーザにスパコンを開放して有効活用できるようにすることなどを想定すると、必ずしも信頼できるユーザだけを想定することは出来なくなるため、システムとして確実なセキュリティを実現することが重要な課題となってくる。

複数の OS・環境を安全に使い分ける手法としては、仮想化技術が近年注目されており、実際に

スパコンで仮想マシンモニタを導入する動きもある。しかし、従来の仮想マシンモニタでは、仮想化によるオーバーヘッドが大きく、スパコンにおけるノード性能を最大限に引き出すという目的には必ずしも適しているとは言えない。一方で、マルチブートなど複数の OS を切り替えて利用する場合、特にユーザが専用 OS を用意する場合などにおいては、ノードの全権限がユーザに渡されることになり、セキュリティを確保することが難しくなってしまう懸念がある。

本研究では、代表者が研究開発してきた仮想マシンモニタ「BitVisor」を応用して、OS に対するオーバーヘッドを極めて低く抑えつつ、必要最小限のセキュリティを確保することにより、多様な用途に応じたノード性能の最適化とセキュリティの確保を両立することを目指す。

近年のスパコンは多数のノードから構成されていることを活用して、一台のノードに複数の仮想マシンを構築するのではなく、ノードの集合の一部を切り出して仮想スパコンとしてユーザに割り当てることにより、それぞれのノードのハードウェアはユーザが直接制御できるようにする。これにより、目的・用途に応じたノードの最適化を可能にする。

一方で、セキュリティを実現するために、仮想マシンモニタによって各ノードのハードウェアへのアクセスを必要最小限だけ監視してアクセス制御をおこなう。これにより、性能とセキュリティの両立を実現する。また、ノードの集合である仮想スパコンの管理機能や独自 OS や大容量のデータを各ノードに高速にデプロイするための枠組みも提供することを目指す。

今年度は、まず、これまでの研究開発において実現してきた OS をスパコンの各ノードに高速にデプロイするためのシステムに関する性能評価をおこなった。従来のシステムでは UDP をベースにした ATA over Ethernet を使用していたが、より安定してストレージ・データを取得できるようにするために、仮想マシンモニタ内部から TCP/IP 通信をできるようにした。また、Network Interface Card(NIC)が 1 つしか搭載されていないマシンでも動作するようにするために、新たに VirtIO NIC に対応することで、ゲスト OS と仮想マシンモニタが NIC を容易に共有して利用することが出来るようにした。

また、スパコン環境における各ノードのセキュリティを確保するための基本的な研究をおこなった。スパコン上でユーザが自由に OS をインストールできるようにすると、管理者権限でハードウェアに自由にアクセス出来るようになるが、この場合ハードウェアに搭載されている不揮発性メモリの内容などを破壊されると、マシンが起動しなくなるなど再インストールでは容易に回復できないダメージを受ける可能性がある。そこで、そのような影響が受ける可能性があるアクセスをいくつか特定したうえで、仮想マシンモニタでそのようなアクセスを防止するための枠組みに関する検討を行った。

また、ライブマイグレーションを実現するための研究開発もおこなった。スパコンのノードのメンテナンスをおこなう場合などに、OS を他のノードに移動する必要が出てくるが、従来の仮想化環境とは異なり、提案環境では OS がハードウェアに直接アクセスしているため、そのままではライブマイグレーションを実現することは難しい。そこで、仮想マシンモニタで物理デバイスの状態を取得・設定することで、ライブマイグレーションを実現するための研究開発をおこなった。

3. 当拠点公募型共同研究として実施した意義

実際にスパコンとして使われていたマシンを利用することで、現実的な環境における評価を実施することが出来た。

4. 前年度までに得られた研究成果の概要

該当なし

5. 今年度の研究成果の詳細

5.1 高速デプロイシステム

本研究では、まずスパコンの各ノードに OS を

高速にデプロイするシステムに関する研究開発をおこなった。このシステムでは、まずユーザが選択した任意の OS をスパコンの各ノードで速やかに使えるようにするために、仮想マシンモニタのレイヤでゲスト OS のネットワークブートをおこなえるようにする。OS カーネルがネットワークブートすることで、各ノードでは新しい OS が速やかに起動することができる。また、その後のストレージへのアクセスに関しては、仮想マシンモニタが捉えてゲスト OS からは透過的にネットワーク越しのサーバへ転送することによって、ゲスト OS を一切改変したり複雑な設定をおこなったりすることなくネットワークブートが実現できる。例えば、Windows や OS X, Linux が修正なしでネットワークブートさせることが出来る。

OS が起動し始めたら、ネットバークブートの場合に発生するネットワーク経由でのストレージへのアクセスのオーバーヘッドを削減するために、バックグラウンドで OS イメージのインストール作業をおこなう。仮想マシンモニタでストレージ領域の管理をおこない、まだ書き込みがおこなわれていない領域に対応するデータをバックグラウンドでサーバから読み込んでディスクに書き込む作業をゲスト OS の動作と平行しておこなう。このとき、ゲスト OS からの書き込み作業と競合しないようにタイミングや順番などを注意しておこなう。いったんインストール作業が完了すると、ストレージへのアクセスは全てローカルディスクへのアクセスとなるため、ネットワークへのアクセスが発生しなくなるため、オーバーヘッドを削減することが出来る。このように、ネットワークブートの利点とローカルブートの利点を併せ持ったシステムを実現することが出来る。

ローカルディスクへのインストール作業が完了した後は、仮想マシンモニタがすることは何もなくなるため、可能な限り脱仮想化の処理をおこなう。脱仮想化とは、デバイスなどへのアクセスを仮想マシンモニタが仮想化・横取りするのを止めて、ゲスト OS が仮想マシンモニタの介在なしに直接デバイスにアクセスできるようにすることである。この脱仮想化を実行時に動的におこなうことによって、ゲスト OS からは気づかれること無くシームレスにオーバーヘッドを削減した状態で動作を継続できるようになる。

従来のシステムでは、ネットワーク上のサーバからストレージのデータを取得するためのネットワーク・ストレージプロトコルとして、UDP をベースとした ATA over Ethernet (AoE)プロトコルを使用していた。しかし、このプロトコルでは再送処理などを自前で実装することが必要であった。現在の実装では、環境によって転送速度が大きく左右されるため、環境に合わせたチューニングが必要となっていた。また、この実装は TCP/IP と同等のことをおこなっており、似たようなものを再実装することには問題があった。

そこで本研究では、新たに TCP/IP に対応したバージョンを研究開発した。TCP/IP のプロトコルスタックとしては lwIP と呼ばれる既存のソフトウェアを活用した。これを仮想マシンモニタに組み込んで動作するようにしたうえで、デプロイシステムと組み合わせることで、ストレージ・データを TCP/IP 経由で取得できるようにした。

しかし、lwIP を用いた実装では、ストレージの転送速度があまり速くできないことがわかった。原因の一つとしては、まず lwIP がもともと組み込み向けであり、構造的にあまり速度を考えて設計されていないというのがある。ほかにも現在の仮想マシンモニタに組み込む際の相性の問題で性能が出ていない可能性もあるが、現在のところ性能低下の詳細な原因は分かっていない。性能低下を抜本的に改善するためには、TCP/IP のプロトコルスタックを根本的に書き直す必要がある可能性もあり、今年度の研究では十分な性能を発揮できるところまでは実装を進めることが出来なかった。

5.2 セキュリティ

本研究では、提案方式をスパコンに適用した場合のセキュリティに関する考察もおこなった。提案方式では、スパコンの各ノードにおいてユーザが自由に選んだ OS をどうさせることが出来るよ

うにすることを想定している。この場合、ユーザがカーネル権限で動作するプログラムを送り込めることになるため、セキュリティに関する配慮が必要になる。

従来の仮想化技術を用いた手法では、仮想マシンモニタが仮想的な OS 実行環境である仮想マシンを作成して、そのなかでユーザの OS が動作することになる。この場合、OS がアクセスできるのは仮想マシンモニタが提供するリソースだけであり、そのリソースは全て仮想マシンモニタによって管理されているため、脆弱性などがない限りは、予め割り当てられた範囲でのアクセスしかすることは出来ない。そのため、セキュリティを担保することは比較的容易である。

しかし提案方式では、スパコン環境においてオーバーヘッドを削減するために、ユーザの OS に対して各ノードのハードウェアに直接アクセスすることを許可している。この方式では、仮想マシンモニタによる介在が少ないため、仮想化によるオーバーヘッドが生じないようにすることが出来る一方、実際の物理的なハードウェアにアクセスされるため、予期しない状況が発生する危険性がある。

例えば、ハードウェアの中には Non-Volatile Memory (NVM)やバッテリ付き CMOS などの不揮発メモリを持っているものがある。これは、各デバイスの設定情報などを保存したりするために用いられるもので、電源が OFF になっても中身のデータが消えずに保持されるという特徴を持っている。この NVM などに格納されている各種設定情報は、デバイスの稼働時の他、起動時の初期化などにも用いられることがある。このような場合、NVM などの値が予期しない不正な値になっていたりすると、デバイスのファームウェアなどが正しく動作せずに、特定のデバイスが動かなくなったり、マシン全体が起動できなくなってしまったりすることがある。いったんこのような状態になってしまった場合には、ソフトウェアのみの手法では容易に復旧することができなくなってしまうことがある。これは、ユーザの OS が誤動作したり悪意のあるユーザによって Denial of Service Attack を受けたりするという点で問題となる。

本研究では、ファジングと呼ばれる手法により、様々な NVM の様々な領域にランダムな値を実際に書き込んでみることで、問題が発生する可能性がある領域を実際に特定した。このような領域には、システムが持つ CMOS 領域のほか、Network Interface Card(NIC) や Graphic Processing Unit(GPU)などのデバイスに含まれる NVM などがある。

このような領域を特定したうえで、仮想マシンモニタによってこれらの領域へのアクセスのみを捉えてアクセスを禁止することで、スパコンのノードに障害が発生することを防止できる。NVM などの領域は、通常動作時にはあまり頻繁にはアクセスしない領域であるため、この部分だけを仮想マシンモニタで捉えるようにしても、仮想化によるオーバーヘッドはほとんど増加しない。

今年度は、実際に Intel Pro1000 のネットワークデバイスに搭載されている NVM 領域への書き込みを禁止する実験をおこなった。具体的には、Z97 というチップセットと連携して動作する I218-V という NIC デバイスを対象として、ゲスト OS から NIC デバイス上にある不揮発性メモリである EEPROM を隠蔽するための実装をおこなった。当該 EEPROM にアクセスする手段としては、NIC を PCI デバイスとみなして内部のレジスタにアクセスする方法と、搭載されたチップセットである Z97 の Serial Peripheral Interface(SPI)を経由してアクセスする方法の２つがあった。これらのアクセスをハイパーバイザで監視して、EEPROM 領域への書き込みを防止する機構の実装を行った。

SPI 経由でのアクセスにおいて、主に使用されるレジスタとしては、Hardware Sequencing Flash Control (HSFC), Hardware Sequencing Flash Status (HSFS), Flash Address (FADDR), Flash Data (FDATA) の４つがある。HSFC は Read, Write, Erase などのアクセス方法や、アクセスの開始、アクセスのサイズなどを指定するレ

ジスタである。HSFS はアクセスの完了やエラーの発生の有無などのステータスを示すレジスタである。FADDR は EEPROM のアクセス先のアドレスを指定するレジスタ、FDATA は読み書きするデータを格納するためのレジスタである。

EEPROM へのアクセスは以下の様な手順で行われる。

(1) FADDR にアクセスするアドレスを指定
(2) FDATA に書き込むデータを格納
(3) HSFC でアクセス方法を指定
(4) アクセス要求を発行
(5) HSFS でステートを確認
(6) ステートが DONE なら終了

本実装では、BitVisor を用いて SPI への I/O アクセスを捕捉する。(3)の段階において I/O の内容を確認し、アクセスが Write や Erase でありかつ FADDR で指定されたアドレスが書き込み禁止の領域であった場合には、(4)の処理がおこなわれないようにする。この時、I/O 要求を遮断するだけだと(5)の段階で HSFS のステートが DONE にならないため、処理が終了せずに OS が終了してしまう場合がある。これを防ぐため、(3)で I/O 要求を遮断した際に、同時に HSFS のステートを DONE かつ ERROR に設定する。

MAC アドレスの読み取り

```
$ethtool -e eth1
Offset      Values
------      ------
0x0000:     d0 50 99 2f 51 f2 01 08 ff ff
```

MAC アドレスの書き換え

```
$ethtool -E eth1 magic 0x15a18086 offset
-0x05 value 0x00
$ethtool -e eth1
Offset      Values
------      ------
0x0000:     d0 50 99 2f 51 00 01 08 ff ff
```

BitVisor による MAC アドレス書き換えアクセスの遮断

```
$ethtool -E eth1 magic 0x15a18086 offset
-0x05 value 0x00
Cannot set EEPROM data:
Operation not permitted
```

実際に動作確認実験を行った結果を図に示す。実験に使用したマシンは、CPU が Intel Core i7-4790K, メモリが 16GB, マザーボードが Z97、カーネルが Linux 3.16.0-38、OS は Linux Mint 17.2 である。動作確認実験では、ethtool を用いて、EEPROM に格納されている MAC アドレスの書き換えを試みた。MAC アドレスを書き換えられると、他のマシンの通信を妨害したり他のマシンへのネットワークパケットを不正に受信したりといったセキュリティ上の問題が生じる可能性がある。

図の一番上のように、本実装を導入していない環境において ethtool を用いて MAC アドレスを読み込むと、d0 50 99 2f 51 f2 という値になっていることが分かる。このとき、図の真ん中のように ethtool を用いて MAC アドレスの領域の offset 0x05 に対して 0x00 という値の書き込みをおこなうと、実際に値が書き換えられて、d0 50 99 2f 51 00 のように最後のバイトが 0 に書き換わってしまっていることが分かる。

一方、提案方式を実装したハイパーバイザ上で動作する Linux から同様の操作を実行すると、図の一番下のように Operation not permitted となって、許可されていない動作として認識され、書き換えを防止できていることが分かる。

マシン AB 間のレイテンシ

また、本手法では NIC に対するアクセスを捕捉するため、ネットワーク性能に対する影響を調べる実験をおこなった。測定にはベンチマークソフトウェア omnitest を使用して、レイテンシの測定

をおこなった。

図に示すように、ベアメタルマシン上（none）の Linux におけるレイテンシと比べて、提案方式（on BitVisor）におけるレイテンシは送信で 3μs、受信は 0.2μs 程度しか増加しなかった。一方、比較対象として、KVM 上で動作する Linux におけるレイテンシを測定すると、送信は 40μs、受信は 25μs 程度もレイテンシが増加した。

実験結果から、提案方式はベアメタル環境における性能をそこなうことなく、Intel Pro 1000 NIC 上の EEPROM に対する書き換え攻撃を確実に防止できていることが確認できた。

5.3 ライブマイグレーション

本研究では、ノードに障害が発生した場合などを想定して、スパコンのノード上で動作する OS をライブマイグレーションする手法に関する研究開発もおこなった。近年のスパコンは多数のノードで構成されており、一つ一つのノードは比較的安価であるため、ノードの故障が発生する可能性がある。ノードが故障すると、そのノード上でおこなわれていた計算結果が失われてしまうため、予期せぬノード故障によるダウンタイムを避ける必要がある。

ノード故障への対応としては、予めスケジュールされたハードウェア交換や、ハードウェアの故障時期を予想する手法などがある。しかし、いずれの場合でも、ハードウェアを停止する必要が有るため、そのままではノード上の OS を停止する必要がある。

これを解決する手法として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は、OS を稼働したまま別の物理マシン上に移動する手法である。これにより、ダウンタイムが発生すること無くハードウェアのメンテナンスをおこなうことができるようになる。

ライブマイグレーションは、仮想マシンモニタによって実現される。従来のライブマイグレーションでは、OS は仮想マシン上で動作しているため、マシンの状態は仮想的に創りだされたものであり、ソフトウェアで容易に移動することが可能である。従って、ライブマイグレーションを実現することはそれほど難しくはない。

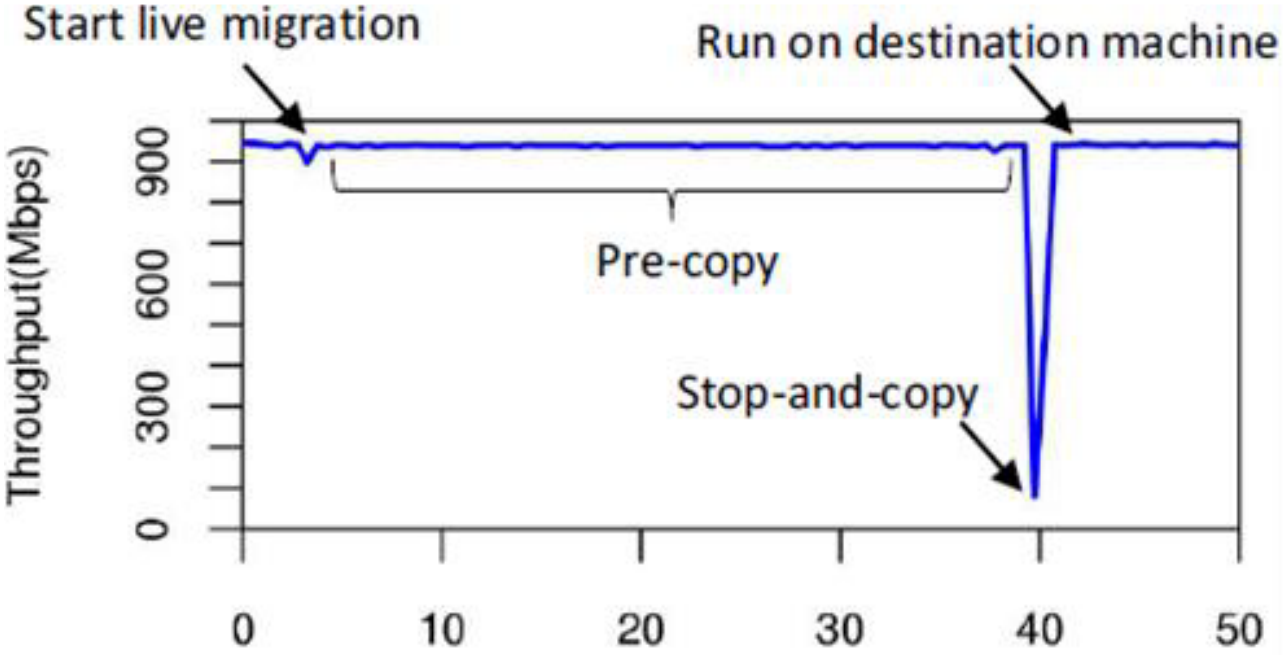
しかし、提案手法では、スパコン環境におけるオーバーヘッドをなるべく削減するために、ハードウェアを仮想化せずに OS に対して直接見せるようになっている。すなわち OS は仮想マシン上ではなく物理マシン上で直接稼働している。この場合、ライブマイグレーションを実現するためには、仮想マシンではなく物理マシンの状態を移動する必要がある。

しかし、物理マシンの状態を取得するのは容易ではない。これは、ハードウェアの中には読み込み専用の状態や書き込み専用の状態が一部に存在しているからである。また、内部に状態を持っている場合もあり、これらの状態をソフトウェアで単純に読み書きして移動することは出来ない。

そこで、我々は間接的な手法を用いて、これらの状態を取得したり設定したりする手法を研究開発した。例えば、NIC に関しては、現在送信しようとしているパケットのバッファを指しているポインタが内部レジスタとして保持されているものがあるが、これらの値はソフトウェアからは直接的には設定できない。そこで、ソフトウェアからダミーのパケットを移送元のマシンに対して送信することにより、間接的にこの値を設定するようにする。

このような間接的に値を取得・設定する手法を幾つかのデバイスに対して実現した。具体的には、RealTek 社製の NIC や Programmable Interval Timer(PIT), Programmable Interrupt Controller (PIC)などである。これらのデバイスに対応することで、最小構成の Linux を物理マシン上で動作させつつ、小さな仮想マシンモニタを用いてライブマイグレーションを実現することが出来るようになった。

実際に実装したハイパーバイザを用いて、マイグレーションをおこなう実験をおこなった。実験に使用したマシンは、CPU が Intel Core i7-4790K、メモリが 4 GB、NIC が Realtek RTL8169 を搭載

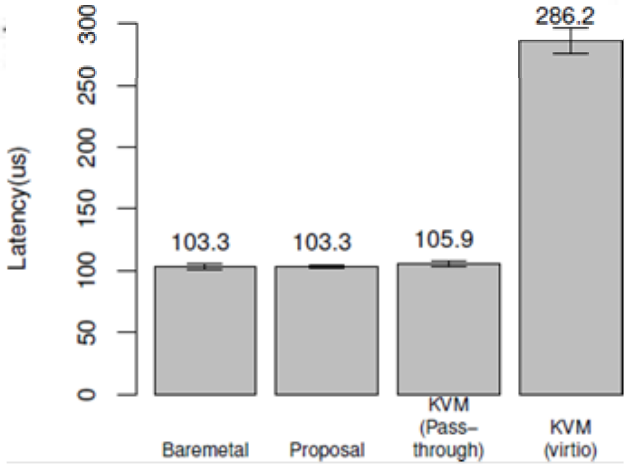


したマシンである。マイグレーションのデータ転送のためには専用に Intel PRO/1000 NIC を使用した。OS は Linux 3.2.65 である。

実験結果は図のとおりである。縦軸は Iperf サーバを動作させ続けた時のネットワークのスループットである。Pre-copy フェーズでメモリを転送している最中であっても、Iperf は動作し続けており、スループットの低下も殆ど無いことが分かる。実験開始から 37 秒ほど経過した時点において、Stop-and-copy フェーズに入って、ゲスト OS をいったん停止させて全ての状態の転送及び設定をおこない、転送先のマシンで動作を再開している。ダウンタイムは 0.7 秒程度に押さえられていることが分かった。マイグレーションの前後において、Iperf のクライアントは通常通り動作し続けており、ベアメタルマシンでのライブマイグレーションが実現できていることが確認された。

次に、通常動作時におけるネットワークに対するオーバーヘッドを測定した。比較対象として KVM も動作させた。

結果を図に示す。”Baremtal”はハイパーバイザのない通常のマシン上におけるレイテンシ、”Proposal”は提案方式のハイパーバイザ上で動作



する場合のレイテンシである。いずれも 103.3μs であり、レイテンシに対するオーバーヘッドは全く無いことが分かる。一方、KVM の場合、virtio を使うと 286.2μs と倍近いオーバーヘッドがかかっていることがわかる。KVM を Pass-through のコンフィグレーションで用いると、105.9μs とオーバーヘッドをかなり抑えられるが、pass-through にしてしまうと仮想デバイスが使えないため、ライブマイグレーションを実現することができなくなってしまう。

このように、実験結果から実装した方式ではベアメタルマシン上でのライブマイグレーションを低オーバーヘッドかつ短いマイグレーション時間で実現できていることが確認された。

6. 今年度の進捗状況と今後の展望

今年度は、主に高速デプロイシステムと、セキュリティ、ライブマイグレーションを実現するシステムに関する研究開発をおこなった。高速デプロイシステムに関しては、当初計画に記載していた TCP/IP 版は動作するようになっているが、性能面での改善をすることは難しかった。今後は、設計や実装を抜本的に見直す必要があると考えられる。

セキュリティに関しても、提案手法においては重要なポイントである。実際に、Intel Pro1000 デバイスを対象にして EEPROM を保護するための機構の設計と実装をおこない、MAC アドレスの不正な書き換えを防止できることを確認した。今後は、BIOS の書き換えなどを防止する手法や、GPU などの電圧調整によってハードウェアが破壊される可能性などに関する研究をおこなっていきたい。

ライブマイグレーションに関しては、順調に研究が進み、国際会議 UCC 2015 における査読付き論文が採択されている。また、当該論文は Best Paper Award を授賞した。今後は対応デバイスを拡大したうえで、論文誌への投稿をおこなっていきたい。

本システム自身のオーバーヘッド削減に関しては、ネステッドページングによるオーバーヘッドが一定程度残っていることが判明している。ネステッドページングは CPU のハードウェアによっておこなわれるものであり、ソフトウェアからは削減は難しいが、CPU が新しくなるたびにそのオーバーヘッドが削減されることが期待されるため、最新の CPU を使うことでオーバーヘッドが軽減されることが期待される。また、ライブマイグレーションのように用途によってはネステッドページングを動的に OFF にすることで、さらなるオーバーヘッドの削減を実現することも目指していきたいと考えている。

7. 研究成果リスト

(1) 学術論文

なし

(2) 国際会議プロシーディングス

[1] Satoru Takekoshi, Takahiro Shinagawa, Kazuhiko Kato. Testing Device Drivers against Hardware Failures in Real Environments. In Proceedings of the 31st ACM Symposium On Applied Computing (ACM SAC 2016), Apr 2016.

[2] Takaaki Fukai, Yushi Omote, Takahiro Shinagawa, Kazuhiko Kato. OS-Independent Live Migration Scheme for Bare-metal Clouds. In Proceedings of the 8th IEEE/ACM International Conference on Utility and Cloud Computing, Dec 2015. ≪Best paper award≫

(3) 国際会議発表

[1] Iori Yoneji, Yushi Omote, Takahiro Shinagawa, Kazuhiko Kato. A new framework for baremetal OS. 6th Asia-Pacific Workshop on Systems (APSYS 2015), Tokyo, Jul 2015.

[2] Satoru Takekoshi, Takahiro Shinagawa, Kazuhiko Kato. Testing Device Drivers against Hardware Failures in Real Environments. 6th Asia-Pacific Workshop on Systems (APSYS 2015), Tokyo, Jul 2015.

[3] Yushi Omote, Takahiro Shinagawa, Kazuhiko Kato. Exit-Less Isolated Execution. 6th Asia-Pacific Workshop on Systems (APSYS 2015), Tokyo, Jul 2015.

(4) 国内会議発表

[1] 東 耕平,竹腰 開,深井 貴明,品川 高廣,加藤和彦.. ベアメタルクラウドにおけるハードウェア保護. 第 136 回システムソフトウェアとオペレーティング・システム研究会. 情報処理学会研究報告, 第 2016-OS-136 巻, 情報処理学会, 2016 年 2 月.

(5) その他（特許，プレス発表，著書等）

なし