様　式　Ｃ－１９、Ｆ－１９、Ｚ－１９　（共通）

科学研究費助成事業　　研究成果報告書

平成　２８　年　　６　月　　１　日現在

機関番号：　１２１０２
研究種目：　挑戦的萌芽研究
研究期間：　2013～2015
課題番号：　２５５４００２２

研究課題名（和文）実時間仮想計算機の研究

研究課題名（英文）Research on real-time virtual machines

研究代表者
新城　靖（Shinjo, Yasushi）

筑波大学・システム情報系・准教授

研究者番号：００２５３９４８

交付決定額（研究期間全体）：（直接経費）　　2,900,000円

研究成果の概要（和文）：従来のホスト型仮想計算機では、実時間アプリケーションを実行することは困難である。本研究では、「仮想計算機ロングポーリング」という新たな仕組みを提供することで、この問題を解決することを提案した。本研究では、提案した仕組みを Linux におけるホスト型仮想計算機モニタ KVM において実装し、その有効性を確認した。提案方式に基づき、ゲスト OS におけるネットワーク入出力を割り込み処理なしで行う事を可能にした。

研究成果の概要（英文）：It is hard to run real time applications in conventional hosted virtual machines. This research overcomes this problem by using the mechanism called virtual machine (VM) long polling. The proposed mechanism has been implemented in KVM, which is a hosted virtual machine monitor in Linux. Using this mechanism, network I/O can be implemented without interrupt handling in a guest operating system.

研究分野：　情報学

キーワード：　オペレーティングシステム　実時間　仮想計算機　ソフトウェア

１．研究開始当初の背景

仮想計算機は、今やシステムの必須の構成要素になった。仮想計算機により、サーバの集約を行ったり、異なる OS（Operating System）用のプログラムを混在させて実行できるようになる。実時間アプリケーションを仮想計算機上で実行できれば便利である。しかしながら、現在のホスト型仮想計算機には、スケジューリング、メモリ管理、および、入出力に実時間アプリケーションを実行する上で問題がある。そのため、実機で実行すれば実時間制約を満たすことが保証できる場合でも、仮想計算機で実行すれば保証できない。

２．研究の目的

本研究の目的は、上記の問題を解決し、実時間アプリケーションを実行できるホスト型仮想計算機を実現することである。研究期間内に、オープンソースで広く使われている仮想計算機モニタ Linux KVM に対して実時間機能を付加する。

仮想計算機で実時間アプリケーションを実行する試みもある。しかしながらその方法では、スケジューリング、メモリ管理、および、入出力に実時間アプリケーションを実行する上で問題があり、実時間制約を満たすことが保証できない。また、実時間とマークされた仮想計算機内で実行される全てのプロセス（非実時間のプロセスを含む）に実時間の高優先度が割り当てられてしまうという問題もある。そのため、ホスト OS や他の仮想計算機に悪影響を与えてしまう。本研究では、このような問題がない実時間仮想計算機を実現する。

本研究では、仮想計算機を用いても実時間アプリケーションを実行できるようにする。具体的にクライアント側では、動画像や音楽を再生するプログラム群を仮想計算機でパッケージ化してインストールを容易にしたり、ホスト OS では対応していないプログラムでも実行できるようにする。サーバ側では、鉄道運行管理等の実時間性が求められるサーバ群を仮想計算機を用いて集約し、消費電力と設置場所を節約できるようにする。

３．研究の方法

本研究では、代表者の独自技術「アウトソーシング」、および、新たに提案する技術「VM ロングポーリング」と「実時間仮想 CPU」を用いて実時間仮想計算機を実現する（図 1）。

アウトソーシングとは、ゲスト OS の高水準のモジュールがホスト OS の高水準のモジュールの機能を積極的に利用することである。従来の準仮想化では、デバイス・ドライバのような低水準のモジュールを置換える。これに対して、アウトソーシングでは、ソケット層やファイル・システム層といった高水準のモジュールを置換える。

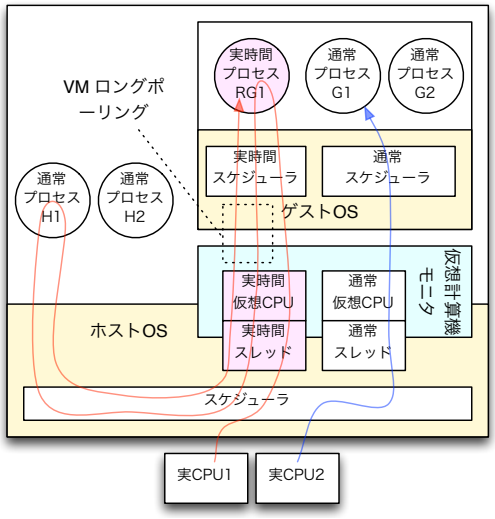

図 1 VM ロングポーリングと実時間仮想 CPU による実時間処理

この時、VMRPC（Virtual Machine Remote Procedure Call）という独自に開発した仕組みを用いる。VMRPC は、分散システムの構築で多く用いられている RPC（Remote Procedure Call）を、仮想計算機環境に特化して実現したものである。VMRPC では、ゲスト OS の高水準のモジュールがクライアント、ホスト OS の高水準のモジュールがサーバとなる。VMRPC は、分散システムの RPC の利点である、機能と意味を手続き呼び出しの形で明確に記述できることを引き継ぐ。VMRPC は、分散システムの RPC とは異なり、共有メモリや vmmcall 命令などの CPU の仮想化支援機能を利用して効率的に実装できる。

本研究では、VMRPC を発展させて、新たに「VM ロングポーリング」という仕組みを導入する。ロングポーリングとは、もともとは分散システムにおいてサーバ側からクライアント側へ情報をプッシュするために考案された技術である。この技術では、サーバは要求を受け付けても、応答を遅延させる。そして、何かクライアントに情報をプッシュしたい時に応答を返す。World Wide Web では、Ajax（Asynchronous JavaScript + XML）という仕組みで複雑なアプリケーションが開発されているが、Ajax にもロングポーリングが使われている。VM ロングポーリングでは、VMRPC において分散システムのロングポーリングと類似のことを行う。

本研究で新たに実現する２つ目の仕組みは、「実時間仮想 CPU」である。これは、仮想計算機に割り当てる仮想 CPU の一種で、ホスト OS では実時間の高い優先度を持つスレッドであり、ゲスト OS では実時間プロセスを専用に実行する CPU に見える。ゲスト OS は、この CPU を使って実時間プロセスを実行する。実行するプロセスがなくなると、次のイベントが発生するまで VM ロングポーリン

グでその仮想 CPU を停止させる。

４．研究成果

本研究では、仮想計算機モニタ Linux KVM、ホスト OS・ゲスト OS 共に Linux を対象として、VM ロングポーリング、実時間仮想 CPU を実現した。まず、仮想計算機モニタ Linux KVM を改変し、仮想 CPU の独立性を実現する。この仕組みにより実質的に ロングポーリングが実現されたことになる。具体的には、カーネル・レベル・スレッドとして実装されているが、仮想 CPU の独立性により、1 つのスレッドがブロックしても他のスレッドは実行を継続できるようにした。次に、実時間仮想 CPU を実現した。具体的には、ホスト OS 側で実時間仮想 CPU に対応したカーネル・レベル・スレッドに実時間処理用の高優先度を付与することで実現した。これらの機能を用いて、Linux のインターバルタイマ機能 hrtimer を実現し、既存の方式と比較して遅延が少ないことを確認した。また、実時間アプリケーションでよく用いられる usleep 機能を、提案方式により実装した。これにより、ゲスト OS においてビデオプレーヤ等のアプリケーションを正確に動作させることが可能となった。実時間ベンチマークプログラム Cycletest を利用して、提案方式の有用性を確認した。

本研究では、提案方式を、入出力の１つ、ネットワーク通信に適用した。従来の仮想計算機では、ネットワーク通信についてはゲスト OS において割り込み処理を利用していたので、その部分において処理時間を精密に見積もることができなかった。これに対して、本研究では、仮想計算機ロングポーリングと実時間仮想 CPU を利用することでゲスト OS における割り込み処理を廃することを可能にした。

提案方式は、Linux において実時間性能を高めるために広く使われている RT PREEMPT Patch をあてたホスト OS において動作している。このパッチは、実計算機では広く使われているが、それをそのまま仮想計算機で使ったとしても実時間性能を高めることはできなかった。提案方式では、ホスト OS において RT PREEMPT Patch をあてることで、ホスト OS が持っている実時間性能をゲスト OS においても利用可能になる。

本研究では、Linux が持っている資源管理機能 cgroups (control groups) を利用することで、実時間仮想 CPU に対応したスレッドとそれ以外の処理を行うスレッドのスケジューリングを効率的に行うことを可能にした。従来は、実時間処理が存在しない時に明示的に CPU 資源を開放しないとシステム全体がフリーズしたように見えることがあった。cgroups を利用することで効果的に非実時間の処理を行い、フリーズを避けることができるようになった。

提案方式の有効性を懸賞するためには、ネットワーク通信の遅延を精密に測定する必要がある。本研究では、このための仕組みを Intel DPDK (Data Plane Development Kit Intel) を用いて実装した。従来の方法では、測定対象にプローブを挿入するために、どうしてもそのオーバヘッドが測定結果に含まれてしまう。この仕組みを利用することで、測定対象となるホストに手を加えることなく、通信遅延を測定できるようになった。

５．主な発表論文等

〔雑誌論文〕(計 5 件)

① Yasushi Shinjo, Wataru Ishida and Jinpeng Wei: "Implementing a parallel world model using Linux containers for efficient system administration", IEEE Second International Workshop on Container Technologies and Container Clouds (WoC 2016), 7 pages (April 8, 2016, Berlin, Germany). http://researcher.watson.ibm.com/researcher/files/us-sseelam/woc-world os-2016-03-22-v27.pdf, 査読有

② 石田航, 新城靖, 佐藤聡, 中井央: "Linux における仮想化技術を用いた世界 OS の実装", 情報処理学会第 26 回コンピュータシステム・シンポジウム (ComSys2015), 11 pages (2015 年 11 月 25 日-26 日). http://id.nii.ac.jp/1001/00145924/, 査読無

③ Ake Koomsin and Yasushi Shinjo: "Running Application Specific Kernel Code by a Just-In-Time Compiler", The 8th ACM Workshop on Programming Languages and Operating Systems (PLOS 2015), 6 pages (October 4, 2015, Monterey, California, USA). http://dx.doi.org/10.1145/2818302.2818305,査読有

④ Ake Koomsin and Yasushi Shinjo: "lua_syscall: Specializing Operating System Kernels by Using the Lua Language", 6th ACM SIGOPS AsiaPacific Workshop on Systems (APSys 2015), 2 pages (2015). http://www.sslab.ics.keio.ac.jp/apsys2015/assets/posters/5.pdf, 査読有

⑤ 石田航, 新城靖, 佐藤聡, 中井央: "仮想化技術による世界 OS の実装の提案", 情報処理学会第 26 回コンピュータシステム・シンポジウム(ComSys2014) ポスターセッション, 2 pages (2014 年 11 月 19 日-20 日). http://www.ipsj.or.jp/sig/os/index.php?plugin=attach&refer=ComSys2014%

2FPoster&openfile=comsys2014_poster_04.pdf,査読無

〔学会発表〕（計１件）

① 新城靖、松下 正吾:"BitVisor におけるゲスト OS・保護ドメイン・USB ドングル間の遠隔手続き呼び出し", 情報処理学会 BitVisor Summit 2（情報処理学会第 25 回コンピュータシステム・シンポジウム併設）(2013 年 12 月 6 日).芝浦工業大学 豊洲キャンパス（東京都江東区), http://www.bitvisor.org/summit2/slides/bitvisor-summit-2-04-shinjo.pdf

〔その他〕

ホームページ等

http://www.softlab.cs.tsukuba.ac.jp/

６．研究組織

(1)研究代表者

新城　靖（SHINJO, Yasushi）

筑波大学・システム情報系・准教授

研究者番号： 00253948

(2)研究分担者

追川　修一（OIKAWA, Shuichi）

筑波大学・システム情報系・准教授

研究者番号： 00271271