協調型仮想計算機におけるホスト呼び出し機構

五明将幸[†] 新城靖[†,‡] 白石光隆[†] 佐藤聡[†] 中井央[†] 板野肯三[†,‡]

筑波大学[†] 科学技術振興機構[‡]

1 はじめに

1 台の PC (Personal Computer) 上で複数の OS (Operating System) を同時に実行するということが広く行われるようになっている．これを実現する方法の一つに VMM (Virtual Machine Monitor) を使う方法がある．VMM とは，仮想計算機を構築・管理するソフトウェアである．VMM はその動作形態により，Type-I と Type-II の 2 種類に分類される．Type-I VMM は実機上で直接動作するのに対し，Type-II VMM は他の OS (ホスト OS) 上のユーザプロセスとして動作する．

仮想計算機の個人利用では，主にホスト OS 上で動作しないアプリケーションを動かすために使われ，セットアップが簡単な Type-II VMM が好まれる．PC 上で動作する代表的な Type-II VMM として，VMware Workstation, Virtual PC, Linux KVM[1] がある．これら既存の仮想計算機は，実機に近い堅牢性を実現するために，ホスト OS とゲスト OS が強く隔離されて動作するように設計されており，ゲスト OS 上からホスト OS 上のファイルやアプリケーションといった資源に直接アクセスする事ができない．このようなゲスト OS とホスト OS が強く隔離されて動作する仮想計算機を，隔離型仮想計算機と呼ぶ．

この隔離型仮想計算機に対して，我々は，個人利用という限定した用途に着目し，ゲスト OS 上からホスト OS 上の資源に直接アクセスできる仕組みを備える協調型仮想計算機を提案している [2]．協調型仮想計算機ではホスト OS 上のファイルやプログラムといった資源をシステムコールやライブラリレベルでアクセスすることができる．本稿では，それを実現する協調型仮想計算機の機能について述べる．

2 協調型仮想計算機の機能

協調型仮想計算機は，ホストコール，シャドウ・プロセス，ホスト・ゲスト間通信の 3 つの機能を提供する．

2.1 ホストコール

VMM は，一般にゲスト OS から VMM の機能を利用するためのハイパバイザコールと呼ばれるインタフェースを提供する．本協調型仮想計算機では，ハイパバイザコールに加え，

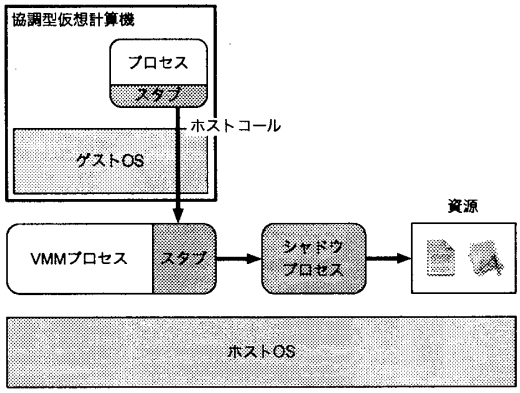

図 1 協調型仮想計算機におけるホスト OS 上の資源へのアクセス

ホストコールを提供する (図 1)．ホストコールは，ゲスト OS 上からシステムコール，およびライブラリのレベルでホスト OS 上の資源を利用するためのインタフェースである．ゲスト OS 上のプロセスは，ホストコールを利用することで，ホスト OS 上のファイルやプログラムにアクセスすることが可能になる．例えば，ホストコールを用いてホスト OS の `fork`, `exec` システムコールを呼び出すことで，ホスト OS 上のプログラムを実行することができる．

2.2 シャドウ・プロセス

本協調型仮想計算機上において，ホストコールを行うプロセスが複数存在した場合，ホスト OS 上での実行コンテキストの整合性がとれなくなる場合がある．例えば，ホストコールを用いてゲスト OS 上からホスト OS 上の/home/john/public_html/index.html というファイルを編集するために，/home/john/public_html に `chdir` し，index.html を `open` することを考える．この時，`chdir` と `open` の間に別のゲスト OS 上のプロセスが/var/www という別のディレクトリに `chdir` してしまうと，本来編集したい/home/john/public_html/index.html というファイルではなく，/var/www/index.html という別のファイルを誤って編集してしまうことになる．

本協調型仮想計算機では，ゲスト OS 上のホストコールを行う主体ごとに，ホスト OS 上でプロセスを生成し，このプロセスがホスト OS 上の資源にアクセスすることで上記の問題を解決する．このホスト OS 上の資源に実際にアクセスするプロセスを，シャドウ・プロセスと呼ぶ (図 1)．シャドウ・プロセスは，協調型仮想計算機を実行したユーザと同じ権限で動作し，カレント・ワーキング・ディレクトリや，オープンしているファイルなどのホスト OS 上における実行コンテキ

The Mechanism of Host OS Calls on a Cooperative Virtual Machine

Masayuki GOMYO, Yasushi SHINJO, Mitsutaka SHIRAISHI, Akira SATO, Hisashi NAKAI, Kouzo ITANO

† University of Tsukuba

‡ Japan Science and Technology Agency

ストを保持する．

2.3 ホスト・ゲスト間通信

ホストとゲストがともにUNIX系のOSを実行する場合，両OS間でパイプの機能を実現できると便利である．本協調型仮想計算機では，両OS間でパイプと同様のプロセス間通信を実現する方法として，現在2つの手法を考えている．

1つ目は，ゲストOSの動的リンク・ライブラリを修正する方法である．`open`, `read`, `write`などのファイル記述子を扱うシステムコールにおいて，対象がホストOS上のファイルであるのかゲストOS上でのファイルであるのかを判断し，ホストOS上のファイルであった場合はホストコールを行うように修正する．

2つ目は，デバイスドライバを用いてゲストOSを拡張する方法である．現在は，/procファイルシステム，または`ioctl()`を利用することを考えている．例えば，/proc/hostpipe/*1*を`open`すると，ホストOS上のシャドウ・プロセスでパイプを生成し，そのパイプに繋がったオブジェクトをゲストOSのカーネル内に生成する．`open`した結果得られるファイル記述子に対して`write`を行うと，そのカーネル内のオブジェクトでホストコールを行い，最終的にシャドウ・プロセスで生成したパイプに`write`システムコールでデータを出力する．この例において，パスに含まれる*1*は，複数`open`される場合にそれぞれを識別するための数である．

2.4 スタブ

本協調型仮想計算機上において，ゲストOSとホストOSの2つのOSを利用するプログラムを**二重OSプログラム**と呼ぶ．二重OSプログラムでは，ホストコールを直接利用することも可能であるが，RPCと同様にスタブが利用できれば，より便利である．

RPCのスタブでは，引数の整列化・非整列化を行うが，本協調型仮想計算機のスタブは，引数の整列化・非整列化を行わない．その代わりに，VMMが確保しているゲストOSとホストOSの両方から参照可能な共有メモリ領域を通じて引数と結果の受け渡しを行う．また，RPCでは最終的にメッセージの送受信を行う通信プリミティブを呼び出すが，本協調型仮想計算機のスタブでは，ゲスト側は，ホストコールを行い，ホスト側は，シャドウ・プロセスを介してシステムコール，またはライブラリ関数を呼び出す．

3 実装

我々は，Linux KVM[1] (Kernel based Virtual Machine)を利用して協調型仮想計算機を構築している．Linux KVMは，Intel VTやAMD-vといったCPUによる仮想化支援機能を使用するType-IIの仮想計算機モニタであり，x86アーキテクチャ上のLinuxで動作する．

本協調型仮想計算機では，Linux KVMが提供するハイパバイザコールを拡張し，ホストコールを実現している．シャドウ・プロセスとホスト・ゲスト間通信は現在実装中である．スタブに関しては，インタフェース定義からスタブを自動的に生成するコンパイラを開発している．また我々は，本協調型仮想計算機を用いてホストOSとゲストOSの両OSのファイルやプログラムを扱う二重OSプログラムである**二重OSシェル**[3]を併せて開発している．

4 関連研究

VMCI[4] (The Virtual Machine Communication Interface)は，VMware Workstationにおいて複数のゲストOSとホストOS間で資源の共有を行うための仕組みである．VMCIは，共有メモリとデータグラムを提供している．一方，本協調型仮想計算機では，共有メモリとホストコールを提供している．これにより，ゲストOS上からホストOS上のファイルやアプリケーションといった資源に，システムコールのレベルでアクセスすることが可能になる．

MacOS X Classic環境は，MacOS XでMacOS 9のアプリケーションを実行するための実行環境である．MacOS X Classic環境では，MacOS 9のAPIをMacOS Xの機能を呼び出すことで実現していると思われる．本協調型仮想計算機では，対象のOSを限定しないため，ホストOSとゲストOSのより柔軟な組み合わせが実現できる．

5 おわりに

本稿では，協調型仮想計算機におけるホスト呼び出し機構について述べた．本協調型仮想計算機では，ホスコール，シャドウ・プロセス，ゲスト・ホスト間通信，およびスタブといった機能を提供することで，ゲストOSからホストOSのファイルやプログラムといった資源へのアクセスを可能にする．

今後は，提案した機能を実装し，機能と性能を評価する．

参考文献

[1] Avi Kivity, Yaniv Kamay, Dor Laor, Uri Lublin and Anthony Liguori: KVM: The Linux Virtual Machine Monitor, *Proceedings of the Linux Symposium*, Ottawa, Ontario, pp. 225–230 (2007).

[2] 五明将幸, 新城靖, 白石光隆, 佐藤聡, 中井央, 板野肯三: ホストOSと協調して動作する仮想計算機の提案, 情報処理学会第19回コンピュータシステム・シンポジウム (ComSys2007) ポスター&デモ・セッション (2007).

[3] 白石光隆, 新城靖, 五明将幸, 板野肯三, 佐藤聡, 中井央: 協調型仮想計算機のための二重OSシェル, 情報処理学会 第70回全国大会 (2007).

[4] VMware: The Virtual Machine Communication Interface, http://pubs.vmware.com/vmci-sdk/ (2007). Accessed 17 December 2007.