Vol.2011-OS-118 No.5
2011/7/27

ADvisor: ゲスト OS の操作に連動した広告を表示するハイパバイザ

小川 夏樹[†1]　大山 恵弘[†1]

情報機器の画面は広告表示先として極めて魅力的である．これまで，広告表示のための様々な方法が既に提案されてきたが，これらは表示場所が制限されていたり，特定の OS やアプリケーションに強く依存しているなどの問題がある．そこで，本研究では，OS やアプリケーションにほとんど依存せず，画面上の任意の場所に任意の広告画像を表示するハイパバイザ ADvisor を提案する．ADvisor は，ゲスト OS および仮想マシンの状態がある条件を満たしたタイミングで，画面上に画像や文字を表示する．例えば，ゲスト OS と I/O デバイスの間で転送されるデータブロックに特定のキーワードが含まれているときに，そのキーワードに連動した広告を表示する．我々は，ADvisor を実装し，それが実行時間に与えるオーバヘッドを，ベンチマークを用いた実験により測定した．

ADvisor: A Hypervisor for Displaying Advertisements Related to Operations on a Guest OS

Natsuki Ogawa[†1] and Yoshihiro Oyama[†1]

Screens of information devices are an extremely attractive platform for displaying advertisements. Although numerous methods for displaying advertisement have been proposed so far, they have problems such as a limited space for displaying and strong dependency on a particular OS or application. In this paper, we propose ADvisor, a hypervisor for displaying arbitrary advertising pictures at arbitrary places on the desktop screen of a guest OS. ADvisor displays pictures and character strings on a screen when a certain condition on a guest OS or a virtual machine is satisfied. For example, when a data block transferred between the guest OS and I/O devices contains a certain keyword, ADvisor displays an advertisement related to the keyword. We implemented ADvisor and measured its runtime overheads through experiments using a benchmark.

1. はじめに

情報通信機器の画面が持つ広告表示先としての価値は高まる一方である．広告表示のための様々な方法が既に存在する．動画サービスの YouTube では，ウェブブラウザ内に表示する動画に広告を重ね合わせる．Apple による iAd は，様々なアプリケーションの画面内に広告を表示することを可能にする．また，Web サイトの画面内やポップアップにバナー広告を入れる伝統的な方法も広く用いられている．

これらの方法には，広告主にとって，いくつかの問題がある．第一は，広告を表示する場所が，動画内やブラウザのウィンドウ内に限られる点である．他のアプリケーションの利用時には広告は表示されない．また，表示されるとしても，ウィンドウ内などの狭い場所に表示せざるを得ないことが多い．第二は，広告を表示させるための仕組みが，OS，ブラウザ，ブラウザアドオンなどに依存することが多い点である．例えば，Apple iAd[1] は iOS に依存しており，Google Adsense[2] はブラウザを起動していないと広告を表示することができない．第三は，広告表示を制限するための設定やソフトウェアにより，ユーザが広告表示を抑制できる点である．例えば，ユーザはブラウザの設定によってポップアップを禁止したり，バナー広告表示を抑制するソフトウェアを使ったりすることができる．

そこで，本研究では，OS やアプリケーションにほとんど依存せず，画面上の任意の場所に任意のサイズの広告を表示させるシステム ADvisor（ADvertising Hypervisor）を提案する．ADvisor は，特定のキーワードが含まれたファイルの保存などのゲスト OS の操作に連動した広告を表示させる．また，ある程度の時間ごとに広告を表示させることもできる．OS のユーザは，たとえ OS 管理者の権限を有していても，ADvisor による広告表示を抑制することができない．ADvisor は広告として，画面上の任意の場所に静止画を表示することができる．また，画面上を移動する画像を表示することもできる．例えば画面左端から出現し画面右端へ消えていくなどの動きを広告につけることができる．ADvisor は広告の表示に限らず，OS のユーザに向けた様々なメッセージを表示する目的にも利用することができる．

ADvisor はハイパバイザである BitVisor[3] を改造することにより実現されている．ハイパバイザとは，仮想マシンを提供し，その上で OS（ゲスト OS）を動作させることを可能に

†1 電気通信大学
The University of Electro-Communications

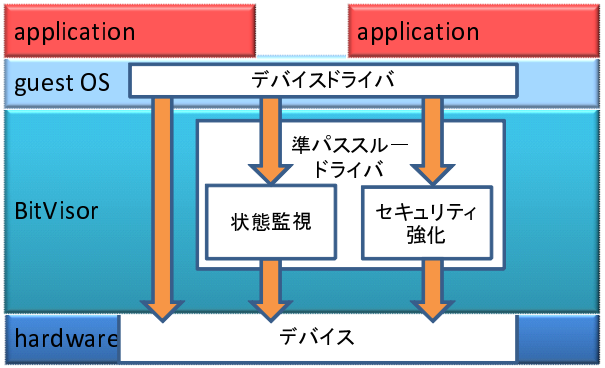

図 1 BitVisor の構造
Fig. 1 The Structure of BitVisor

するソフトウェアである．著名なハイパバイザとしては Xen[4] や VMware ESX[5]，KVM[6] がある．BitVisor は軽量なハイパバイザであるので，低オーバーヘッドでの広告表示の実現が期待できる．ADvisor はゲスト OS とは独立に表示処理を実行するので，OS のユーザが利用中のアプリケーションに関係なく，確実に画面に広告が表示される．ADvisor はグラフィックハードウェアのフレームバッファ領域を更新することにより，ゲスト OS による画面描画処理とは無関係に，ゲスト OS の画面上に画像を表示させる．

本論文は以下のように構成されている．2 章では，ハイパバイザである BitVisor について述べ，3 章で提案システムについて説明する．4 章では実験の結果を示しそれについての議論を行う．5 章では，関連研究について述べる．6 章で今後の展開について示し，7 章で論文をまとめる．

2. BitVisor

BitVisor とは，筑波大学などによって開発された，ハードウェア上で直接動作する軽量なハイパバイザである．ゲスト OS は同時に 1 つのみ動作する．BitVisor は，準パススルー型と呼ばれる方式を利用する．準パススルー型とは，ゲスト OS からハードウェアに可能な限り透過的にアクセスさせ，セキュリティ機能の実現のために，最低限必要なアクセスのみを仮想マシンモニタで捕捉する方式である．一部のアクセスのみを捕捉することにより，ストレージやネットワークの暗号化などのセキュリティ機能を実現することができる．**図 1** に示すように，仮想マシンモニタで捕捉する必要があるアクセスには，制御 I/O とデータ I/O の 2 種類がある．制御 I/O は，デバイスによるデータ転送を制御するための I/O で，転送するデータの場所やアクセス方法，データ転送の開始，終了などを指定する．データ I/O は，実際にデータ転送を行う I/O である．仮想マシンモニタが制御 I/O を捕捉してアクセスの状態を把握し，データ I/O を捕捉してデータを取得，更新することにより，ゲスト OS から透過的に暗号化や復号化などのセキュリティ機能を実現できる．BitVisor は，仮想化支援機構（Intel VT または AMD Virtualization）をサポートする Intel アーキテクチャの CPU 上で動作する．

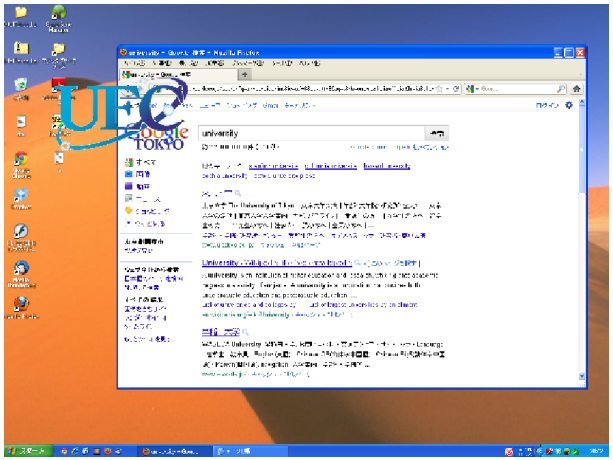

図 2 ADvisor 上で動作する Windows XP のスクリーンショット
Fig. 2 A screenshot of Windows XP running on ADvisor

3. 提案システム

3.1 システム概要

ADvisor はゲスト OS の下で動作し，仮想マシンを管理している．ADvisor は，ゲスト OS および仮想マシンの状態がある条件を満たしたタイミング（後述）で，**図 2** のように画面上に広告の画像や文字を表示させる．図 2 は，電気通信大学のロゴがゲスト OS である Windows XP の画面上に表示されている状態である．

ADvisor の利用シナリオを以下で述べる．

インターネットカフェの PC への広告表示　インターネットカフェで提供される PC において，ADvisor とゲスト OS はカフェの店員がインストールし，客には一般ユーザ権限だけを渡す．カフェは，ADvisor の導入にあたり広告主や広告代理店と契約を結び，インストール数や稼働時間などに応じた収入を得る．

シンクライアント端末への広告表示　クラウドサービスとして用いられるシンクライアントの画面上に ADvisor による広告を表示する．サービスを提供する業者は，サーバマシンに ADvisor をインストールし，顧客に利用させる OS をその上で動作させる．ユーザは仮想デスクトップ環境を無料もしくは割引料金で得られる代わりに，その仮想デスクトップに ADvisor による広告が入る．顧客には OS の管理者権限を与えることができる．

組織内での情報周知　組織のある部署のシステム担当者がその部署の全メンバーの PC に ADvisor とゲスト OS をインストールする．ADvisor の管理者権限はシステム担当者が保持し，ゲスト OS の管理者権限は各メンバーに渡す．システム担当者は，全メンバーに周知したい情報を ADvisor によって表示する．例えば節電，P2P ソフトウェアの利用禁止，部署内の会合の告知などのメッセージを表示させることができる．

ADvisor の管理者と，ゲスト OS のユーザや管理者は異なることを仮定している．ADvisor の管理者は広告を見せようとする人であり，ゲスト OS のユーザや管理者は広告を見る人である．

ADvisor は BitVisor のバージョン 1.0 を改造して実装されている．ADvisor の概要を**図 3** に示す．BitVisor は元々，ゲスト OS による画面制御関連の操作を捕捉しない．すなわち，ゲスト OS は，自身が持つデバイスドライバによってグラフィックハードウェアを制御する．それは ADvisor においても同様であり，ゲスト OS は直接グラフィックハードウェアを操作して画面を描画する．ただし，ADvisor も適切なタイミングでグラフィックハードウェアを操作し，ゲスト OS が表示する画面の一部を広告で上書きする．

ADvisor はマッチング部と描画部からなる．マッチング部では，デバイスに書き込まれるデータと，特定のキーワードとの間のマッチング処理を実行する．キーワードにマッチするデータが書き込まれる場合には，そのキーワードに関連する広告の表示を描画部に要求する．描画部では，ゲスト OS の画面に広告を表示するための処理を実行する．

3.2　広告の表示処理

3.2.1　画像情報の更新

画面情報の更新は，グラフィックハードウェアが持つフレームバッファを ADvisor が直接書き換えることによって実現する．これは幾世らによる研究[7] の方法によって実現している．具体的には以下の流れで画面情報を書き換える．まず，PCI ハードウェアから設定情報を読み取り，グラフィックハードウェアの制御レジスタ領域および VRAM 領域のアドレスを得る．この情報はハードウェアの所定の I/O ポートに I/O を行うことにより得る．VRAM 領域の中には，画面情報が入っているフレームバッファ領域が存在する．次に，その制御レジスタ領域への読み書きを通じて，VRAM 領域の中からフレームバッファ領域のアドレスを得る．その後，フレームバッファ領域のメモリを直接書き換えることにより，画面に画像や文字を表示する．現在，ADvisor は Intel 945GM チップセットのオンボードグラフィックハードウェアを対象に実装されている．

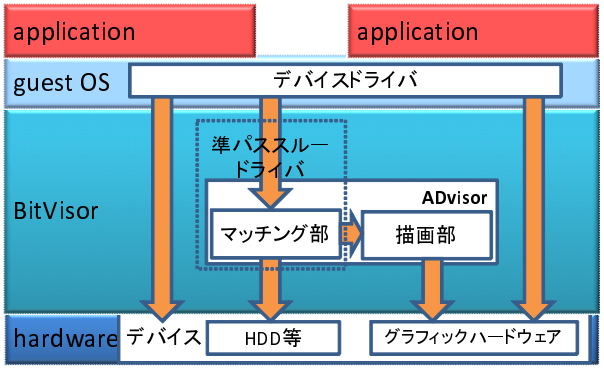

図 3　ADvisor の概要
Fig. 3　Overview of ADvisor

現在の実装では，画像を表示したいピクセルに対応するフレームバッファ領域内のメモリアドレスに，表示したい点の色情報を 1 点ずつ書き込んでいく．グラフィックハードウェアが提供する高度な描画機能は現在利用していない．この処理はグラフィックハードウェアにのみ依存し，ゲスト OS に依存しないため，多様なゲスト OS 上に広告を表示できる．

ADvisor による画面情報の更新は，ゲスト OS による画面情報の更新を全く考慮せずに行われる．よって，ゲスト OS と ADvisor とで更新処理が競合する可能性はある．例えば，広告が上に表示されたアプリケーションのウィンドウを移動させると，広告の画像の一部がウィンドウと一緒に移動して切れるなどの現象が発生する．この場合でも，画面表示が崩れるなどの視覚面での影響があるだけであり，OS がクラッシュするなどの深刻な障害は発生

しない．その理由は，ADvisor はグラフィックハードウェアのフレームバッファ領域を更新するだけで，ゲスト OS が描画に用いているデータ構造は更新しないためである．

3.2.2 移動しない画像の表示

ADvisor は指定の位置に指定の画像を表示させることができる．現在は，ADvisor のビルド時に管理者が表示位置と画像をあらかじめ決めておく必要がある．ADvisor は自らの広告を消す処理は実行しない．OS のユーザは広告の上にウィンドウなどを新たに表示することにより広告を消すことができる．しかしその消去方法は面倒でありユーザの利便性を下げるため，今後，別の方法で広告を消す処理を組み込むことを検討している．

逆に，ユーザが広告を消せないようにすることは，ADvisor を拡張すれば可能である．最も単純にそれを実現するには，非常に短い時間間隔で繰り返し広告を表示すればよい．より優れた方法としては，フレームバッファ領域の広告表示部分に相当するページを書き込み禁止にしておき，ゲスト OS によるそのページへの書き込みをハイパバイザが捕捉するというものがある．

3.2.3 移動する画像の表示

移動する画像は，画像を一定の時間間隔で少し離れた場所に描画することにより表示できる．その際，新しい画像を描画する前に，前の画像を消す必要がある．そのため，ADvisor は画像を描画する前にその領域の背景情報を保存しておき，画像を消す際には保存しておいた背景情報を用いる．画像を消すには，ゲスト OS が所持している本来の画面情報を広告画像に上書きするのが理想であるが，ADvisor はその情報を取得できないためこの方式をとっている．

ただしこの方式を単純に実装すると一つの問題が生じる．それは，広告の画像の表示中に，その画像上に別のウィンドウなどが上書きされた場合には，保存しておいた情報を書き戻すと，その領域に古い情報が表示されてしまうという問題である．そこで，広告の画像上に上書きされた部分については，背景画像を書き戻さないようにする．上書きされた部分を知るには，背景を書き戻す際に広告画像と現在の画面の同じ位置のピクセルをそれぞれ比較する．一致しなかった場合，そのピクセルは更新されたと見なし，背景画像を書き戻さないようにする．

3.3 画像を表示するタイミング

広告を表示するタイミングとしては，現在 2 種類が実装されている．

3.3.1 一定数の処理の実行

この方式では，ある程度の時間経過ごとに広告の画像を表示する．これを実現するには，ADvisor に一定の時間が経過したことを認識させる必要がある．ADvisor では，時間経過をタイマハードウェアを用いて取得するかわりに，実装の単純さから CR3 レジスタの更新情報を用いている．すなわち，CR3 レジスタが一定回数更新されることを広告表示のタイミングとして利用する．CR3 レジスタはゲスト OS のプロセスコンテキストスイッチが行われる際に更新されるので，適切に回数を設定すれば，ある程度の時間間隔での広告表示が期待できる．

3.3.2 ゲスト OS の操作との連動

ADvisor は，ゲスト OS の操作に連動してそれに応じた広告を表示する．具体的には，ゲスト OS のデータ I/O を捕捉してマッチングすることで行う．

ADvisor の利用者は，キーワードと画像の組の集合をあらかじめ ADvisor に登録しておく．登録されたキーワードがゲスト OS とデバイスの間で転送された時には，その文字列に関連づけられた広告の画像が画面に表示される．例えば，キーワードを「university」，それに関連した画像を電気通信大学のロゴとしておく．すると，university という文字列が含まれたファイルがディスクに保存される際に，ゲスト OS のデスクトップに電気通信大学のロゴが表示される．

広告として表示させる画像データとフォントデータおよびキーワードは，ADvisor のソースコードに ADvisor の管理者が埋め込む．管理者はそのソースコードをビルドして ADvisor を得る．ソースコードに情報を埋め込んでビルドするのは煩雑であるにもかかわらず，この方法を採用しているのは，ADvisor はハードウェアの上で直接動作しておりゲスト OS 以外の OS を持たないことから，上記のデータをファイルの形で保持することが難しいためである．

ゲスト OS とデバイス間の転送には，ATA デバイス，ネットワークデバイス，USB ストレージデバイスによるデータ I/O がある．また，BitVisor では，PIO (プログラム I/O)，メモリマップド I/O，DMA (Direct Memory Access) の 3 つの転送方式によるデータ I/O アクセスが捕捉できる．ADvisor は，それらすべての方式により ATA デバイスに書き込まれる I/O データに対してキーワードマッチングを行う．ADvisor は，データ I/O の値が実際にストレージに書き込まれる前に溜まるデータ（すなわちディスクブロック）に対してキーワードマッチングを行う．

3.4 画像を表示する位置

ディスプレイのどこに画像を表示するかは重要な問題である．表示場所によっては，ユーザの作業の邪魔になってしまうためである．現在は，ADvisor のソースコード内での指示

に従って，画像が固定的な場所に表示されたり，固定された経路を移動するようになっている．今後 2 つの拡張を ADvisor に導入することを検討している．

1 つ目は，ADvisor が自動的に適切な位置を判定して画像を表示する手法である．例えば，デスクトップ上のウィンドウが表示されていない場所に画像を表示する．ピクセルの色の配置や更新頻度をチェックすることで，ユーザにとって重要な場所かを判定し，あまり重要でない位置に画像を表示する．

2 つ目は，画像を移動させるための指示をユーザが出せるようにする手法である．例えば，ユーザが画像を任意の場所に動かせるようにする．しかし，ADvisor はハイパバイザ層においてゲスト OS に依存しない形で実現されているため，それを実現するのは単純ではない．まず，マウスカーソルの位置などのゲスト OS が管理する情報を取得することが出来ない．また，カーソルキーなどのキー入力はゲスト OS に対する入力として利用されるため，ハイパバイザへの指示に利用すると，ユーザの意図しない動作を招く．そこで，現在導入を検討している手法は，ユーザは画像を移動させたり消去したい時には，ADvisor が定めたコマンド文字列をディスクに書き込むというものである．ADvisor は監視しているデータ I/O の中にコマンド文字列を検出した場合には，表示中の画像にそのコマンドを適用する．例えば，ユーザは画像を左に動かしたいときは「moveleft」などの文字列を適当なファイルに書き込んで保存する．

4. 実　　験

本章では，ADvisor を用いた実験とその評価について記す．まず，動作確認の実験について述べる．次に，ADvisor の実行時オーバーヘッドについて述べる．実験環境は以下の通りである．

- CPU: Intel Pentium D 3.00 GHz
- chipset: Intel 945GM
- memory: 2 GB

4.1 動作確認

ADvisor が期待通りの動作をするか，Windows XP，Ubuntu 8.04 を用いて確認した．図 2 は，ADvisor 上で動作する Windows XP の画面である．ウェブブラウザ上で打ち込んだ「university」という文字列は，あらかじめ登録しておいたキーワード群に含まれるため，その文字列に関連づけられた電気通信大学のロゴが表示されている．**図 4** は，同じく Ubuntu 8.04 での動作画面である．テキストエディタに打ち込んだ「air-conditioning」と

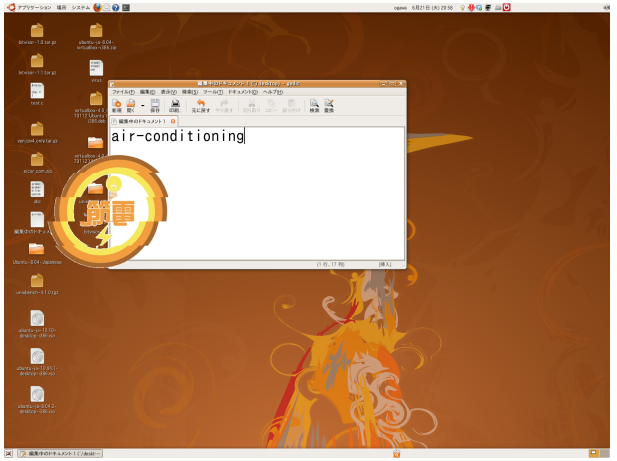

図 4 ADvisor 上で動作する Ubuntu 8.04 のスクリーンショット
Fig. 4 A screenshot of Ubuntu 8.04 running on ADvisor

いう文字列がキーワードにマッチして，節電のロゴが表示されている．ロゴは画面左から右へ移動している状態である．ロゴが歪んで見えるのは，画像が移動している状態でスクリーンショットを取得したためである．

4.2 オーバーヘッド

ADvisor を使用することによるゲスト OS のオーバーヘッドを測定するために，Unixbench 4.1.0 を用いて性能比較を行った．OS は Ubuntu 8.04 を用いた．比較した環境は次の通りである．

- no hypervisor: OS を実機上で動作させた環境
- BitVisor: 変更されていない BitVisor を OS の下で動作させた環境
- ADvisor: ADvisor を OS の下で動作させた環境

結果は**図 5** のようになった．図 5 のグラフの縦軸はベンチマークの結果として出力される Index を表す．Index は性能の指標であり，高ければ高いほど性能が高いことを意味する．いくつかの項目では，ADvisor や BitVisor を動かした状態と，no hypervisor の差は大きいが，ADvisor と BitVisor の差は小さい．特に，単純な計算を行うようなベンチマークでは，差はほとんどない．ただし，ファイルコピーを行うベンチマークでは，ADvisor は BitVisor と比べて性能が若干低下することが確認された．この原因は，後者のベンチマー

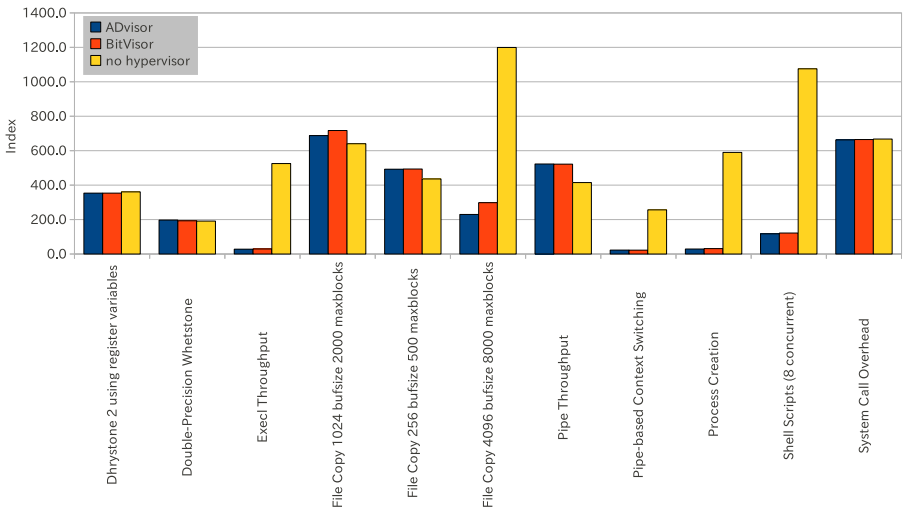

図 5 Unixbench の結果
Fig. 5 Unixbench result

クではディスクへの書き込み処理が行われる回数が多く，その結果，キーワードマッチングの処理も頻繁に実行されたためであると考えている．

5. 関 連 研 究

ユーザの操作と連動した既存の広告表示システムは様々なものがある．

ウェブブラウザ上でのシステムにおいて代表的なのは，Google Adsence，Amazon associate[8] といった検索連動型広告である．これらはユーザの検索や開いているウェブページにマッチした広告を表示するが，ウェブブラウザ上でのみ動作する．一方，ADvisor では任意のアプリケーションがディスクに書き込むデータに含まれる文字列に連動して広告表示が行える．

モバイル向けの広告システムとして，Apple iAd がある．多種多様な表示手段が用意されており，ユーザの操作に連動した広告を，使用中のアプリケーションから離れることなく視聴することができる．iAd は OS に組み込まれている点で ADvisor と異なる．ADvisor は Windows，Linux を含む多様な OS 上に広告を表示することができる．

Mei らの研究[9] ではビデオ向けのオーバレイ広告について，中沢らの研究[10] ではインターネット向け動画広告について改良方法を述べている．また，Li らの研究[11] では，ゲーム向けオンライン強制広告についてのシステムを提案している．これらの研究はいずれもアプリケーション層からのアプローチをとっている点で本研究とは異なっている．

BitVisor におけるデータ I/O のマッチングを行った研究として，Giang らの研究[12] がある．Giang らの研究では，マッチングをマルウェア検出のために行っているが，本研究では広告のための単語マッチングを行っている点で異なっている．

6. KVM への移行

今後，現在広く使われている KVM に ADvisor を移行することを検討している．BitVisor を用いたのは，実機上で OS を動かすのに近い速度でゲスト OS を動かせるためである．ただ，複数のゲスト OS が立ち上げられずホスト OS も持たないため，広告やキーワードの情報を与えるための作業が煩雑であるという問題もある．さらに，グラフィックハードウェアのメモリを直接書き換える現在の方法はハードウェアへの依存が大きい．KVM は，複数のゲスト OS を動すことができ，ホスト OS を持つので画像の動的な追加などがしやすい．また，ハードウェア依存の問題も解決できる．現在実装を始めており，KVM の仮想ディスプレイ上に移動しない画像を表示させることに成功している．今後，画像の移動やユーザの操作に連動した広告表示に取り組んでいく．

7. まとめと今後の課題

ユーザが入力する文字列に連動した広告を画面上に表示するハイパバイザ ADvisor を提案した．ADvisor は，BitVisor を改造して実装されている．現在，あらかじめ登録された単語がディスクに書き込まれるたびに決まった画像を表示させる機能を有する．我々は ADvisor を用いた実験を行い，ユーザの入力に連動した広告を表示させることが可能であることを確認した．また，ADvisor 導入によるプログラムの実行時間の変化を測定した．

今後の課題は，画面表示位置の改良や，ネットワークなどのディスク以外のデバイスによるデータ I/O への対応対応である．さらに，KVM にシステムを移行し，画像の動的な追加や，マッチングアルゴリズムの改良，グラフィックハードウェア依存の問題の解決などに取り組む予定である．

謝辞 本研究は科研費（23700032）の助成を受けたものである．

Vol.2011-OS-118 No.5
2011/7/27

参 考 文 献

1) Apple iAd, http://advertising.apple.com/.
2) Google AdSense, https://www.google.com/adsense/.
3) Shinagawa, T., Eiraku, H., Tanimoto, K., K. Omote, S. H., Horie, T., Hirano, M., Kourai,K., Oyama,Y., Kawai,E., Kono,K., Chiba,S., Shinjo,Y. and Kato.,K.: BitVisor: A Thin Hypervisor for Enforcing I/O Device Security., *In Proceedings of the 5th ACM SIGPLAN/SIGOPS International Conference on Virtual Execution Environments (VEE 2009)*, pp. 121–130 (2009).
4) Barham,P., Dragovic,B., Fraser,K., Hand,S., Harris,T., Ho,A., Neugebauer,R., Pratt,I. and Warfield,A.: Xen and the art of virtualization, *In Proceedings of the nineteenth ACM symposium on Operating Systems Principles (SOSP19)*, pp. 164–177 (2003).
5) Waldspurger,C. A.: Memory Resource Management in VMware ESX Server, *In Proceedings of the 5th Symposium on Operating Systems Design and Implementation (OSDI '02)* (2002).
6) Kivity,A., Kamay,Y., Laor,D., Lublin,U. and Liguori,A.: kvm: the Linux Virtual Machine Monitor, *In Proceedings of the 2007 Ottawa Linux Symposium (OLS '07)*, pp. 225–230 (2007).
7) 幾世知範, 平野学, 品川高廣, 奥田剛, 河合栄治, 加藤和彦, 山口英：仮想マシンモニタ BitVisor のためのロールベースアクセス制御機構の設計と実装, 情報処理学会研究報告, Vol. 2010-CSEC48, No.15 (2010).
8) Amazon Associate, https://affiliate-program.amazon.com/.
9) Mei,T., Guo,J., Hua,X.-S. and Liu,F.: AdOn: toward contextual overlay in-video advertising, *Multimedia Systems*, Vol.16, No. 4-5, pp. 335–344 (2010).
10) 中沢実, 池田康, 中野朋紀, 服部進実：イベント駆動型動画広告配信システムの提案と実装, マルチメディア通信と分散処理研究会報告, Vol. 2004, No.22, pp. 157–162 (2004).
11) Li,L., Mei,T. and Hua,X.-S.: GameSense: game-like in-image advertising, *Multimedia Tools and Applications*, pp. 145–166 (2009).
12) Giang,T.T.D., 大山恵弘, 忠鉢洋輔, 品川高廣, 加藤和彦：準パススルー型 VMM のマルウェア検出機能による拡張, 第 22 回コンピュータシステム・シンポジウム (ComSys 2010), pp. 13–20 (2010).